AMIGA DOS

DAS MAGAZIN FÜR AKTIVE AMIGA-ANWENDER

DM 6,50
sfr 6,50
öS 52

AMIGA DOS

5/90

5/90
ISSN 0937-2717
DMV-Verlag

Viele Tips und Tricks

- Linkchecker
- Modula-2
- CLI und BASIC für Einsteiger

DTP

Schweres Gerät im Praxistest

Music Power wie noch nie

TFMX Soundeditor

Profi Genlock

Das Studio zu Hause

Top-Listings

Superschneller Fraktalgenerator

Komfortabler Filemonitor

Test

Deluxe Video

Citizen Swift 24





Desktop-Video im Aufwind

Die Zeiten, in denen die Erstellung professioneller Videos Fernsehstudios mit ihrem millionenschweren Equipment vorbehalten waren, sind vorbei. Der Amiga hat längst Einzug auch in diese Studios gehalten und vermag Gerätschaften zu ersetzen, die noch vor einigen Jahren den durchschnittlichen Fünf-Jahres-Lohn eines Angestellten verschlungen hätten. Dies bedeutet für den Heimanwender, daß auch zu Hause jetzt Projekte in Angriff genommen werden können, deren Realisierung vor einigen Jahren schlichtweg unmöglich gewesen wäre. Die Preise für benötigtes Zubehör wie Genlocks, RGB-Splitter und Digitizer, aber auch für entsprechende Software haben einen Stand erreicht, der jedem die Chance gibt, für wenig Geld seine Ideen in die Tat umzusetzen. Bei allem Enthusiasmus über diese Möglichkeiten sollte man allerdings den Zeitaufwand nicht außer acht lassen. Wer sich über kleinere, einfach zu realisierende Projekte zunächst Sicherheit im Umgang mit der Materie verschafft, kann viele entstehende Probleme schon im Vorfeld klären und hat so wahrscheinlich größere Aussichten, auch spektakuläre Videos zu erstellen. Apropos spektakulär: Besonders möchten wir Sie auf den Fraktalgenerator in dieser Ausgabe hinweisen, der nicht nur durch seine Farbenpracht beeindruckt. Außerdem ist er der zur Zeit schnellste Fraktalgenerator, der bis dato in einer Zeitschrift für den Amiga veröffentlicht wurde. Auf vielfachen Wunsch haben wir auch noch einmal die Problematik der Linkviren unter die Lupe genommen. Unsere Tips-und-Tricks-Ecke bietet Ihnen eine wirksame Waffe gegen die ungeliebten Eindringlinge, denen schon manche Diskette zum Opfer gefallen ist. Selbstverständlich hat AMIGA DOS noch wesentlich mehr zu bieten, egal ob Sie gerade erst einsteigen oder sich schon länger mit dem Amiga beschäftigen. Doch überzeugen Sie sich selbst. Sicherlich können wir nicht alle Fragen im Rahmen unseres Heftes klären, Hilfe ist jedoch nur einen Telefonanruf weit entfernt. Bleibt uns nur noch, Ihnen viel Spaß bei der Lektüre der neuen AMIGA DOS zu wünschen, denn uns hat es viel Freude gemacht, dieses Heft für Sie zusammenzustellen.

Herzlichst Ihr

Markus Matejka

Markus Matejka

INHALT

NEWS & TRENDS

SOFTWARE

HARDWARE

TIPS & TRICKS

KURS



Bild 1. Logikspiele erfreuen sich großer Beliebtheit. Quadrant 500 kann Ihnen da so manche harte Nuß zum Knacken aufgeben.

Seite 73

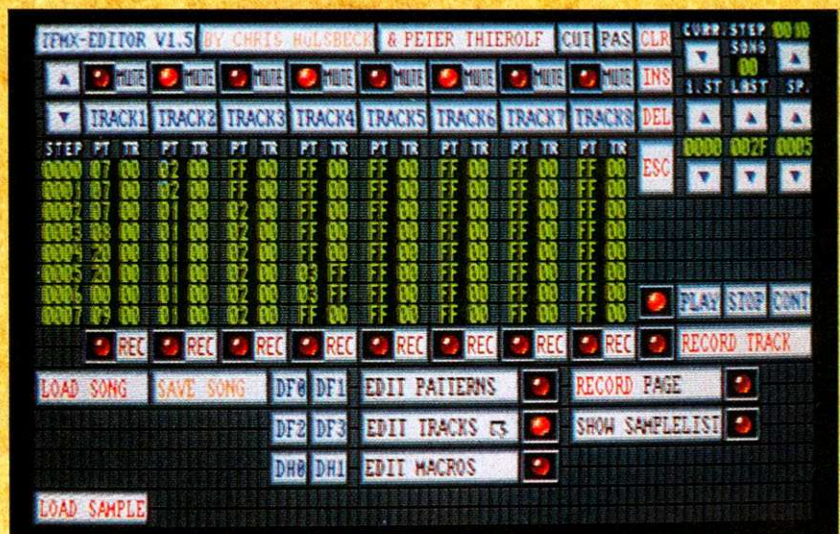


Bild 2. TFMX – ein Soundeditor, der mit vielen Möglichkeiten aufwarten kann. Was TFMX leisten kann, lesen Sie auf

Seite 26



Bild 3. Hoyle Official Book of Games beinhaltet einige der beliebtesten Kartenspiele der Welt

Seite 142

LISTING

WERKSTATT

PUBLIC DOMAIN

SPIELE

RUBRIKEN



Bild 4. Die DTP-Anlage von Commodore ist in Verbindung mit entsprechender Software eine DTP-Alternative. Lesen Sie den Praxisbericht zur Arbeit mit DTP auf

Seite 18

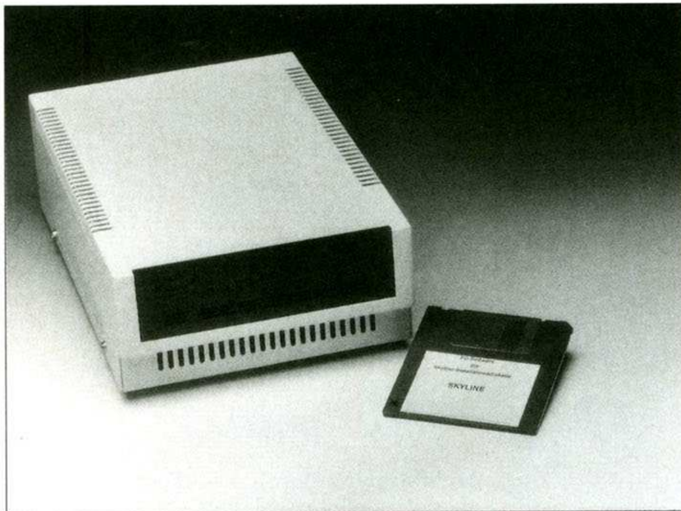


Bild 5. Begleiten Sie uns in die faszinierende Welt der Fraktale – mit unserem Super-Listing Fraktalgenerator, dem schnellsten Fraktalgenerator, der je in einer Zeitschrift veröffentlicht wurde

Seite 80

Skyline-Autoboot-Festplatte für A500/1000

Die Firma Skyline bietet eine neue Autoboot-Festplatte für den Amiga 500 und den Amiga 1000 mit neuer grafischer Oberfläche sowie benutzerfreundlicher Menüführung an. Diese Festplattenlösung beinhaltet ein 5,25-Zoll-Harddisklaufwerk, Kabel, Kleinteile sowie entsprechende Software zum Autoboot. Vier unterschiedliche Größen von 20, 30, 40 und 60 MByte sind durchgängig mit einem Omti-Controller versehen.



Skyline Autoboot-Festplatte

Neues von NORDIC POWER

NORDIC POWER, das Freeze-Modul (getestet in der AMIGA DOS 04/90) besitzt seit März 1990 einige neue Eigenschaften. Hier sind sie:

1.) Eine eingebaute Dia-Show ermöglicht es, jede Art von IFF-Grafiken, die mit NORDIC POWER 'eingefroren' und abgespeichert wurden, als Komplett-Vorführung wiederzugeben. Dabei kann es sich auch um Interlace-Bilder handeln. Die Bilder müssen dazu nur auf im Amiga-Format formatierten Disketten vorliegen. Die Reihenfolge der Bildwiedergabe, den automatischen Ablauf und die manuelle Bilderwahl kann man einstellen.

2.) Der Cheat-Modus ist ebenfalls neu, besser bekannt als Poke-Finder. Es ist jetzt also möglich, im Speicher des Amiga automatisch nach Werten zu suchen, die die Anzahl der Leben repräsentieren. Ebenfalls ist es möglich, jetzt Trainer-Pokes aus Zeitschriften ohne Probleme einzugeben und zu nutzen.

3.) Ein Compacker befindet sich jetzt ebenfalls im Modul, wobei die Möglichkeit besteht, zwischen den verschiedenen RAM-Größen und 'PACK' zu wählen. Allerdings sollte sich außer dem 'gefreezten' Spiel nichts anderes im Speicher befinden.

Die vor dem März gekauften NORDIC der S-Serie (SC oder SP) können gegen einen Aufpreis von 29,– DM zuzüglich 10,– DM Versandkosten aufgerüstet werden. Im Lieferumfang sind eine Einbauanleitung für das Update und die Anleitungsergänzung enthalten.

Info: Data & Electronics
Venlo B.V.
Postbus 3119
NL-5902 RC Venlo
Niederlande

Version 5.0

Neuigkeiten sind auch von der Firma Manx Software Systems Inc. zu vermelden. Seit Beginn dieses Jahres liefert Manx die neueste Version ihres Aztec C-Compilers aus.

50-MHz-Board von Intelligent Memory

Für den Amiga 2000 hat die Firma Intelligent Memory ein neues Board entwickelt: Hurricane 2800 MK II-50.

Eine erste Vorstellung dieses Produkts fand am 9.2.1990 in Paris statt. Anhand eines Raytracing-Bildes, das in einer fantastisch hohen Geschwindigkeit aufgebaut und umgesetzt wurde, konnte man die Leistungsfähigkeit dieses Boards bereits erahnen. Das Topmodell Hurricane 2800 MK II-50 wird demnächst mit einer MC 68030 RC-50 CPU erhältlich sein. Dieses Board läuft mit einer Frequenz von 50 MHz und erreicht eine relative Geschwindigkeit, die rund 20mal schneller als der Standard-Amiga ist.

Der Preis dieses Produkts wird bei zirka 4500 DM inklusive CPU und weiteren Unterlagen liegen. Es kann mit bis zu 16 MByte dynamischen RAMs aufgerüstet werden. Das Board ermöglicht auch ein schnelles SCSI-Interface, das zur Steuerung jeglicher Standard-SCSI-Festplatten verwendet werden kann, und ist voll autobootfähig unter Kickstart 1.3. Ein Performance-Test mit dem SCSI-Port am Hurricane 2800 MK II-50 ergab Schreib/Lese-Zugriffsgeschwindigkeiten bei 550 kByte bis zu 500 Directory-Scans pro Sekunde und rund 200 Create/Close pro Sekunde, bezogen auf das Standard-Festplatten-Testprogramm "DPERF2". Der Ronin-Sieve-Benchmark-Test, der in der Amiga World 4'90 veröffentlicht wurde, ergab im Vergleich zu einem ähnlichen 68030-Board folgende Werte: Ein Ablauf mit einer ständigen Frequenz von 33 MHz lief beim "normalen" 68030-Board in 4,25 Sekunden ab, wohingegen das Hurricane 2800 MK II-50 genau 3,00 Sekunden benötigte. Weitere Beispieldaten zu diversen Tests:

- Zum Zeichnen eines Kreises (100mal) wurden 2,00 Sekunden benötigt.
- Sys-Partition-Speedtest lief bei 2500 Iterationen rund 0,84 Sekunden.

Wir werden Sie über weitere Einzelheiten in einer der nächsten Ausgaben näher informieren.

Eine umfassende Übersicht

...über aktuelle Fachliteratur und Software- bzw. Hardware-Produkte rund um den Amiga hat der Technik Support Verlag im Auftrag der Firma Commodore herausgegeben. Der Amiga Katalog 90 umfaßt rund 320 Seiten, kostet 20,- DM und ist ab Mitte März in Warenhäusern, im Fachhandel und im Buchhandel erhältlich. Hier steht nicht die Quantität, sondern die Qualität der Produkte im Vordergrund. Die Palette reicht von neuesten Publikationen zum Amiga über brandaktuelle Themen wie DTP oder Desktop Video bis hin zu den neuesten Entwicklungen im Hardwarebereich wie Turbo Boards etc.

Termine * * * Termine * * * Termine

• BaCom
Zum dritten Mal findet vom 19. bis zum 21. April im Baden-Badener Kongreßzentrum die südwestdeutsche Büro- und Computerfachmesse statt. Besondere Beachtung findet das Rahmenprogramm "Digital Design", wo neueste Präsentationen aus der Welt der Computergrafik und -animation vorgestellt werden.

• MECOM
In Saarbrücken findet in der Zeit vom 18. bis zum 20. Mai die MECOM statt.

• BASCOM
Basel ist Treffpunkt der Computer-Freaks in der Zeit vom 09. bis 13. Mai.

• Summer CES
Die internationale Summer Consumer Electronics Show, kurz CES genannt, findet in der Zeit vom 2. bis zum 5. Juni in Chicago statt.

'En Passant' – die Schach-Datenbank

Für Freunde des königlichen Spiels ist es oft wichtig, sich auf anstehende Spielbegegnungen (Turniere, Meisterschaften oder auch nur zu Hause) vorzubereiten. Aber es ist oft auch zeitraubend, wenn man erst eine Menge Bücher durchsehen muß, bis man an bestimmte Spielzüge oder Spielbegegnungen kommt oder sie schlimmstenfalls gar nicht erst findet.

Von der Firma Habermehl und Sander gibt es zu diesem Zweck ein neues Amiga-Programm namens 'En Passant'. Hierbei handelt es sich um eine Schach-Datenbank, in welcher der Schachspieler eine Vielzahl an Partien, Zügen und Wissen findet. Bei den Partien kann eine Analyse herangezogen werden, in der es möglich ist, eigene Züge der Probe halber einzubringen und das Ergebnis zu sehen. Sind die eigenen Züge nicht erfolgreich, kann der Spieler zur Ausgangsstellung zurückgehen und einen neuen Versuch starten.

Das Programm läuft auf allen Amiga mit mindestens 1 MByte Speicher und einem Laufwerk. Festplatten werden vom Programm unterstützt, so daß die Datenbank schnell bereit ist. Im Lieferumfang sind Programmdiskette, Datendiskette mit über 600 Großmeisterpartien sowie ein ausführliches deutsches Handbuch enthalten.

Info:
Habermehl & Sander
Tannenweg 11
3575 Kirchhain 1
Tel.: 06422/7222



'En Passant' – die Schach-Datenbank

Westernheld am Bildschirm



Der Westphaser – Schießen elektronisch

'Westphaser' – so nennt sich eine Lichtpistole von Loriciel, mit der man auf den Bildschirm schießen kann, ohne Bruchstücke zu hinterlassen. Im Westphaser ist ein lichtempfindlicher Halbleiter-Baustein eingebaut, der Lichtreflexe vom Bildschirm in elektrische Signale umwandelt und diese über die serielle Schnittstelle an den Amiga übergibt. Dem Paket liegt ein Action-Schießspiel bei, das eine fröhliche Ballerei im Wilden Westen zum Inhalt hat. Auf der Diskette befindet sich ebenfalls ein Beispielprogramm in Amiga-BASIC zum Einlesen der Informationen über die Schnittstelle. Weitere Spiele für den Westphaser sollen bald erscheinen. Zu beziehen ist der Pseudo-Revolver bei

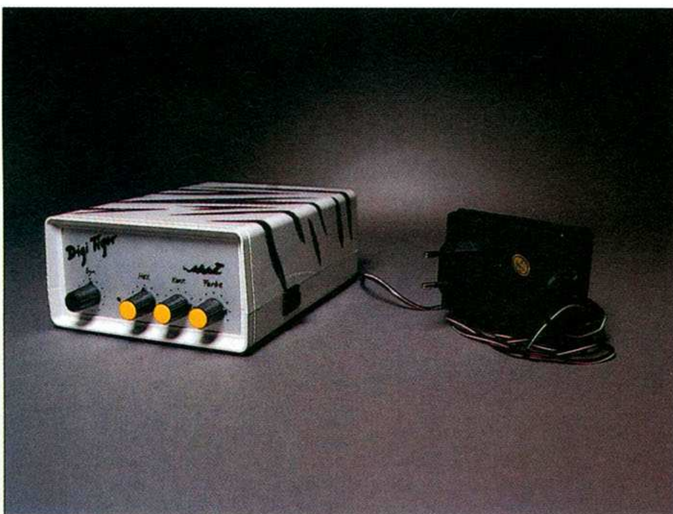
Rushware
Bruchweg 128
D-4044 Kaarst
Tel.: 02101 / 6070

DigiTiger – Video Digitizer V1.0

Die Firma Klaus-D. Tute aus Hannover hat ein neuartiges Video-Digitizer-System auf den Markt gebracht. Mit dem eingebauten RGB-Splitter und dem SP+-Bildspeichersystem ist es möglich, qualitativ hochwertige Digitalisierungen von Schwarzweißbildern und Farbvorlagen von Videoquellen relativ schnell und auch problemlos zu erstellen. Die Bedienung erfolgt über Menüs oder Tastatureingaben und ist durch die deutsche Programmführung vollkommen unproblematisch. Das Speichern und Laden der Bilder erfolgt im Standard-IFF-Format wahlweise mit Icon. Mit einer zusätzlichen Umschaltbox (die leider nicht im Lieferumfang enthalten ist) können die Bilder sofort ausgedruckt werden. Das Programm bietet eine Auswahl zwölf verschiedener Bildauflösungen einschließlich Overscan. HAM-Bilder können mit 4096 Farben in 20 bis 45 Sekunden übertragen und speicherfertig berechnet werden. Die Digitalisierungszeiten liegen bei rund 2 Sekunden pro Bild in Schwarzweiß und 20 Sekunden in Farbe bei freier Einstellung von Helligkeit, Kontrast, Farbsättigung, Farbton und Schärfe. Das Programm ist auf allen Amiga-Geräten mit 10 von 12 Auflösungen schon ab 512 kByte Speicher (HAM-Interlace ab 1 MByte) voll lauffähig.

Eine Demodiskette ist für 10,– DM bei der Firma Tute erhältlich und ermöglicht, alle Funktionen zu testen. Der DigiTiger kostet einschließlich Software, Kabel, Netzteil und 50seitigem deutschem Handbuch 698,– DM.

Klaus D. Tute
Mathildenstr.12
3000 Hannover 91
Tel.: 0511/629825



DigiTiger im Blickpunkt

Im Blickpunkt: GREMLIN

Die Firma Gremlin Graphics Software Limited hat einige neue Spiele angekündigt.

RAMROD

Ein Spiel, bei dem man die Wahl zwischen zwei recht unterschiedlichen Charakteren hat: Auf der einen Seite Ramrod, der muskelbepackte, egoistische Charakter, dessen Name für den Titel Pate stand. Auf der anderen Seite Rambot, ein nett anzusehender blau-metallic-glänzender Roboter. Danach steigt man in das Geschehen ein, genauer gesagt in ein dreidimensionales Terrain mit vielen Klippen, Kanälen und Hügeln. Dort kann man dann mit seinem gewählten Charakter auf Punkte- und Münzjagd gehen. Stellt man sich auf kleine glänzende Scheiben, kann man zu fast jedem beliebigen Ort teleportiert werden. Mit dem Fahrstuhl gelangt man direkt in den Arcadenteil – einem Spiel im Spiel – wo man sich in vier unterschiedlichen Disziplinen pro Level austoben kann. Jedes einzelne muß erfolgreich beendet werden, um ins nächste Level zu gelangen. Einige Stellen sind sehr schwierig zu erreichen, hier helfen dann sogenannte Luftlöcher weiter. Fällt man in eines hinein, dämpfen sie den Fall. Steigt man auf eines auf, können sie einen emporheben.

Eines ist besonders wichtig: Ramrod oder auch Rambot müssen ständig in Bewegung gehalten werden, da sonst Lebensenergie verlorengeht. Es gibt eine Vielzahl unterschiedlicher Bonusarten: "Speed", "Bopper", "Super 8" oder "8 way" – rundum alles, was ein Held wie Ramrod zum Bestehen seiner Abenteuer benötigt.

Die vier Levels sind sehr abwechslungsreich gestaltet und fördern damit gleichzeitig den Spielspaß. Level 1 ist das High-Tech-Level, wo die Oberfläche durch Computer-erstellte Bilder auffällt. In die Zeit der industriellen Revolution fühlt man sich bei Level 2 – dem Low-Tech-Level zurückversetzt. Level 3 ist das Aztec-Level, wo unter anderem der Tempel des Quetzalcoati und dessen heiliger Altar zu finden sind. Anfang und Ende bildet das Fantasy-Level, auch Fantastec genannt. Kurzum ein interessantes Spiel im 3D-Format, das für den Amiga ab März 1990 im Handel erhältlich ist.

Combo Racer

Eine Simulation für Rennen von Motorrädern mit Seitenwagen, mit optionalem Ein- oder Zwei-Spieler-Modus. Spielt man mit zwei Spielern, übernimmt der zweite die Kontrolle des Beifahrers. Das Spiel verfügt über acht unterschiedliche Rennschauplätze mit den verschiedensten Szenarien. Daneben ermöglicht ein spezielles Feature, der Track-Editor, sich seinen eigenen Rennkurs zu gestalten. Ein Spiel für Rennsportfreunde, das 3D-Routinen enthält und für den Amiga ab April im Handel zu erwerben ist.

Skidz

In den Straßen der Großstadt ist man mit einem BMX-Rad oder dem Skateboard unterwegs. Das Ziel: der schärfste Typ in der Stadt zu sein. Insgesamt sind sieben Levels in sieben Tagen zu bewältigen. Dabei ist man auf den Straßen, den Parks, im berüchtigten Chinatown und an anderen Schauplätzen unterwegs. Ein zusätzlicher Arcadenteil vervollständigt das Spiel.

Jeder Parcours muß erfolgreich beendet werden, um zum nächsten Level zu gelangen. Jeder Ort verfügt über eine Vielzahl von Gefahren, die zu überwinden sind. Es ist entscheidend, wie man in der Zeit, die man zur Verfügung hat, lernt, mit dem Rad oder dem Skateboard umzugehen. Am besten ist, man verdient etwas Geld, in dem man Verbrecher fängt, und bessert mit diesem Geld seine Ausrüstung im Laden auf. Mit einer perfekten Ausstattung kann man dann die verschiedensten "Stunts" ausprobieren, bevor man sich ins Gewühl stürzt. Denn dort hat man es mit Räubern, Katzen und Hunden, radfahrenden Tramps oder auch Tauben zu tun. Skidz, ein Spiel, das neugierig macht, ist ab März für den Amiga erhältlich.

Ultimate Golf und Shark Attack

Ultimate Golf ist eine realistische Golfsimulation, die auf Originalschauplätzen und reellen Informationen basiert. Hier sind alle Features eines natürlichen Golfplatzes wie beispielsweise diverse Bunker, Bäume, Sträucher und Hügel so weit es geht naturgetreu auf einem dreidimensionalen Terrain umgesetzt worden. Interessant sind digitalisierte Sequenzen einzelner Schläge des bekannten Golfers Greg Norman, einer Koryphäe auf diesem Gebiet. Daneben besteht die Möglichkeit, auf den einzelnen Kursen umherzugehen. Ein computerisierter Caddy hilft mit seinen Ratschlägen, die Wahl des Schlägers zu entscheiden. Insgesamt siebzehn verschiedene Eisen stehen dem geübten Golfer zur Verfügung, dazu noch jede Menge unterschiedlicher Effekte. Besonders zu erwähnen: Ultimate Golf und Shark Attack sind im Doppelpack zu bekommen, das heißt, zwei Spiele sind zum Preis von einem und ab sofort für den Amiga zu einem Preis von £ 24.99 zu erhalten.

IMPOSSAMOLE

Monty, der Maulwurf, ist zum x-ten Male seit 1987 wieder unterwegs.

Er liegt unter einer Palme am Strand und harrt der Dinge, die da kommen. Doch plötzlich ... fällt ihm eine Kokosnuß auf den Kopf. Er läuft ziellos umher, ohne den merkwürdigen Schatten zu bemerken, der über ihm schwebt. Ein greller Blitz, und Monty findet sich in Gestalt zweier Schattenfiguren wieder. Die geheimnisvolle Umwandlung kann beginnen. Monty ist urplötzlich Impossamole, der allgegenwärtige, weise Maulwurf der 90 Jahre, dessen Zuhause die Straße ist.

So beginnt ein neues Abenteuer, das unser kleiner Freund Monty bestehen muß. Folgen Sie Monty auf seinem Weg durch das Labyrinth von Tunneln und Höhlen in den insgesamt vier unterschiedlichen Levels. Drei Arten von Waffen stehen Monty zur Verfügung: Laser, Bomben und Kugeln. Begleiten Sie ihn bei seinem gefahrvollen Abenteuer, es dürfte sicherlich interessant werden. Impossamole wird ab April für den Amiga im Handel erhältlich sein.

Federation Quest One – BSS Jane Seymore

Vor zehn Jahren verließ das große Forschungsraumschiff Jane Seymore mit einer Mission die Erde. In den Tiefen des Orion-Arms der Galaxie sollten unbekannte Welten nach Lebensformen erforscht werden. Ein falsch berechneter Hypersprung brachte dann die Katastrophe. Der Wiedereintritt erfolgte zu nahe einem Neutronenstern, so daß das Schiff in eine tödliche Strahlung geriet. Der größte Teil der Crew starb oder wurde so krank, daß sie das Gedächtnis verloren und keine Ahnung mehr hatten, welche Aufgabe sie an Bord zu bewältigen hatten. Das Überlebenssystem des Schiffes wurde bei dem defekten Hypersprung schwer beschädigt, so daß die Überlebenden kaum eine Chance hatten, dies zu reparieren, um nach Hause zurückkehren zu können. Viele Lebensformen konnten entkommen und besetzten das Schiff. Ein Funkspruch konnte jedoch noch abgesetzt werden.

Viele Lichtjahre davon entfernt im Hauptquartier der Föderation wurde dieser Funkspruch aufgenommen und eine Crew zur Rettung des havarierten Schiffes gebildet.

Ihre Aufgabe ist es, als Kommandant des Rettungsschiffes, die BSS Jane Seymour zu orten, zu reparieren, die entlaufenen Lebensformen zu fangen und das Schiff wieder nach Hause zu bringen. Das Spiel vereinigt eine Menge unterschiedlicher Features von Strategie bis Rollenspiel in sich. Die Szenarien erscheinen jeweils im dreidimensionalen Blickwinkel. Eine Statusanzeige gibt Auskunft über Gesundheit und Umgebung des Spielers. Über 30 verschiedene Alienformen können Ihnen die Mission ebenso erschweren wie defekte Roboter oder die "verrückte" Crew an Bord. Das Spiel erstreckt sich über zwanzig verschiedene Levels, die jeweils über ein Passwort zu erreichen sind. Daneben steht dem Spieler und auch seinen Gegnern ein großes Arsenal an unterschiedlichen Waffen zur Verfügung. Man bewegt sich in über 100 Räumen auf drei verschiedenen Ebenen des Schiffes. Ein Spiel für lange anhaltenden Spielspaß. Federation Quest One für den Amiga ist ab März im Handel erhältlich.

Amiga-Grafik hochwertig ausgedruckt



Amiga-Bilder hochwertig ausgedruckt...

So schön man mit dem Amiga zeichnen und malen kann, so schlecht sieht das ausgedruckte Ergebnis meistens aus. Querstreifen ruinieren jedes noch so sorgfältig erstellte Bild.

Eine Abhilfe bietet die Firma CGD Dr. Buddemeier an. Wer keinen Farbdrucker mit hohen Qualitäten besitzt oder die entworfenen Grafiken professionell nutzen möchte, kann die Vorlagen an die Firma CGD senden. Diese werden mit hochwertigen Tintenstrahl- und Thermotransferdruckern streifenfrei auf Papier gebracht. Eine Größe bis DIN A4 ist dabei möglich, bei Tintenstrahldruck auch größer.

Kostenlose Infos mit Druckbeispielen können angefordert werden bei

CGD Dr. Buddemeier
Schlesienstr. 40
D-4400 Münster
Tel.: 0251 / 62214



... mit Tintenstrahl- und Thermotransfer-Druckern

'Gefrorene' Programme

Das 'Amiga Action Replay' ist ein neues Freezer-Modul, das über eine Vielzahl von Möglichkeiten verfügt. Der Speicher des Amiga kann ausgelesen, enthaltene Programme können in gepackter Form auf Diskette abgespeichert werden. Das Modul erzeugt auf der Diskette ein eigenes Format, auf dem bis zu drei Programme abgespeichert werden können. Die Programme können hinterher auch ohne Modul wieder geladen werden.

Ein Trainer sucht in Spielen die Werte für 'Leben' heraus und ermöglicht eine Änderung. Der enthaltene Sprite-Editor bringt Sprites aus Programmen auf dem Bildschirm, wo sie geändert werden können. Zum Schutz vor Viren enthält das Modul einen Virusdetektor, mit dem die bekanntesten Viren erkannt werden können. Bilder und Musik können separat abgespeichert werden, eine Zeitlupenfunktion ermöglicht das langsamere Ablaufen von Programmen.

Ein 68000-Assembler-Disassembler ermöglicht das Einsehen in Quelltexte und das Editieren von Quelltexten.

Das 'Amiga Action Replay' kostet 189,– DM zuzüglich Versandkosten und ist zu beziehen bei

Eurosystems
Hühnerstr.11
D-4240 Emmerich
Tel.: 02822/45589

Samurais Edelwaffe

'Sword of the Samurai' heißt ein neues Action-Abenteuer-Rollenspiel von Microprose. Der Spieler wird hierbei in das alte Japan entführt, in dem noch einzelne Länder existieren, die sich gegenseitig um die Vormachtstellung streiten. In der Rolle eines Samurai muß der Spieler für Ruhe und Ordnung sorgen, und dies unter anderem durch aussagekräftige Kampfszenen.

'Sword of the Samurai' ist bis jetzt zwar nur als PC-Version erschienen, eine Amiga-Version wird jedoch aller Voraussicht nach ebenfalls bald erhältlich sein.



Sword of Samurai: Kämpfen wie ein echter Samurai

Mehr Speicher für den A 500

'ERAM MEGA' und 'Megamodul' heißt eine RAM-Kombination für den Amiga 500, die diesen mit einem zusätzlichen Speicher bis zu 2,3 MByte aufrüstet. Das Besondere daran ist, daß der zusätzliche Speicher nach und nach eingebaut werden kann. 'ERAM MEGA' bildet dabei die Grundstufe. Auf der Platine befinden sich neben einer akkugepufferten Uhr 512 kByte RAM in Form von MBit-Chips. Als nächstes kann dann das 'MEGAMODUL' zusätzlich eingebaut werden; es enthält in der Grundversion noch einmal 512 kByte. Dieses Modul läßt sich nun nacheinander in der folgenden Konfiguration mit MegaBit-Chips aufrüsten: 1,5 oder 2 oder 2,3 MByte. Beide Platinen verfügen über vergoldete Anschlußstecker, sind abschaltbar und werden mit deutscher Einbauanleitung ausgeliefert.

Die Preise für 'ERAM MEGA' und 'MEGAMODUL' sehen so aus:
ERAM MEGA 199,– DM,
MEGAMODUL 250,– DM (Grundversion), 375,– DM (auf 2 MByte, nur mit Kickstart 1.3), 489,– DM (auf 2,3 MByte).

Info: Tröps + Hierl
Computertechnik GmbH
Jordanstr.3
D-5040 Brühl
Tel.: 02232 / 45018

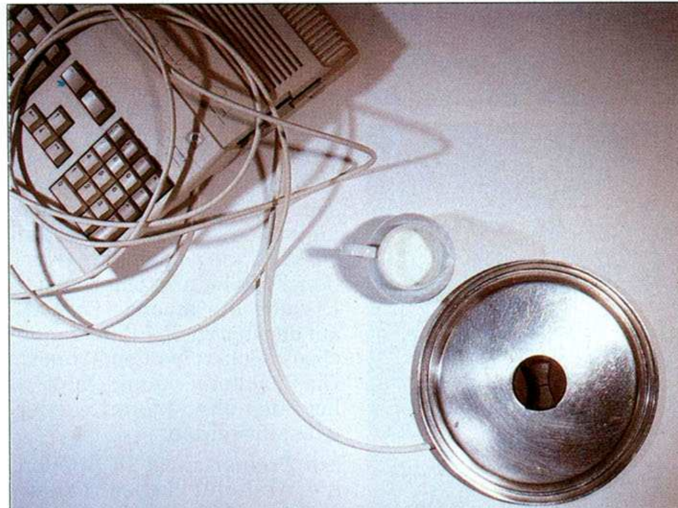


Bild 1. Die benötigten Teile: Ein Kochtopfdeckel und Mäusemilch, das Koaxialkabel paßte leider nicht aufs Bild.

Schüsseltechnologie

Amiga als Privat-TV-Empfänger

Die Redaktion der AMIGA DOS war überrascht – Satelliten-TV auf dem Amiga-Monitor durch den Amiga selbst? Etwas Ungeheuerliches schien wahr geworden zu sein, denn die Nachricht einer bisher unbekannten Firma für Computerzubehör, den Amiga mit wenigen Mitteln in einen Satelliten-Receiver zu verwandeln, erschien uns zu gewagt. Doch sehen Sie selbst.

Zum Umbau eignet sich allerdings nur der A500 durch seine kompakte breitflächige Bauweise, dies sei schon mal vorangestellt.

Wie jedoch geht nun das Ganze vonstatten?

AMIGA DOS sprach mit Gutbert-Ronald Unz, dem Chef der G.R.Unz-Antennenbau GmbH:

AMIGA DOS: Herr Unz, wie sind Sie auf die Idee gekommen, ein Umbauset für den Amiga herauszugeben, welches diesen befähigt, Satellitenprogramme auf handelsüblichen Monitoren auszugeben?

G.R.Unz: Eigentlich war es ein Unglücksfall, der mich auf die Idee brachte. Beim Bau meines Wohnhauses legte ich naturgemäß mit Hand an, Sie wissen schon, man hat Elektriker gelernt und kann damit ganz schön Kosten sparen... Nun, beim Einrichten meines Computerraums hatte ich den Amiga aus meiner alten Wohnung mitgebracht und fing abends nach Einbruch der Dunkelheit an, Steckdosen für die einzelnen Geräte zu setzen. Leider konnte ich die Sicherung nicht herausnehmen, da ich sonst im Dunkeln gestanden hätte. Es wäre ja auch nicht weiter tragisch gewesen, wenn meine Frau, die leider etwas tölpelhaft ist, ääh, das sollten Sie jetzt bitte nicht abdrucken (bitte streichen, die Red.), nicht just in dem Augenblick mit einer frischen Gemüsesuppe hereingekommen und über das Netzteil des Amiga gestolpert wäre. Ich versuchte, den Kochtopf zu fassen, bekam jedoch nur den Deckel, leider aber auch das offene Stromkabel zu fassen. Mit der anderen Hand lag ich gerade auf dem Abschirmungsblech meines A500, den ich aus Reparaturgründen geöffnet hatte.

Nun, so geschah's! Der Strom floß über den Kochtopfdeckel durch meine linke Hand, trat am linken Ohr mit einem Lichtblitz aus, der wiederum führte um meinen Hinterkopf herum, drang am rechten Ohr ein, floß als normaler Strom zu meiner rechten Hand und gelangte von da aus in das Abschirmblech. Im gleichen Moment hörte ich den Schrei meiner Frau: "Da, RTL Plus!" Leider sprach im gleichen Augenblick die Sicherung an, so daß es stockdunkel wurde. Trotzdem behauptete meine Frau steif und fest, Kabelfernsehen gesehen zu haben.

Den eigentlichen Versuch wollte ich nicht noch einmal wiederholen, aus verständlichen Gründen wohlgemerkt, aber es mußte einen Weg geben, diesen Vorgang noch einmal zu erzeugen. Und dann fiel es mir wie Schuppen von den Haaren...

ADOS:...von den Fischen...

G.R.Unz:...mir doch egal! Also plötzlich wußte ich, was geschehen war. Der Deckel war die Satelliten-Schüssel, der Lichtkranz um meinen Hinterkopf der Verstärker. Ich begann zu experimentieren und schon bald hatte ich die komplette Schaltung entwickelt!"

Soweit G.R.Unz. Wir, die Redaktion der AMIGA DOS, haben es geschafft, die Erlaubnis für einen Abdruck der Schaltung zu erhalten. (Bild 1).

SAT 1 und zufrieden

Was passiert nun eigentlich im Rechner? Zuerst ein Hinweis: Weil man niemandem zumuten kann, durch 220 Volt aus der Steckdose einen Lichtkranz um den Hinterkopf zu erzeugen, ist es notwendig, für entsprechenden Ersatz zu sorgen. Für eine gleichwertige Verstärkung benötigt man auf den Zentimeter genau Koaxialkabel mit einer Länge von 2869,73 Meter. Dieses Kabel wird mit etwas Lötzinn und einem Lötkolben an den Kochtopfdeckel gelötet und zwar so: Seele (der Mitteldraht) an den Griff, die Abschirmung an den Rand. Als Kochtopfdeckel eignen sich nur Einmachtopfdeckel mit einem Mindestdurchmesser von 2,23 m. Der Kochtopfdeckel muß nun in einem Abstand von 10,73 Metern vom Amiga aufgestellt werden, die restlichen 2859 Meter werden um den Amiga in einem Umkreis von 82 cm als Spule gelegt. Daraus ergibt sich eine Spulenhöhe von 7,43 Metern. Durch Einschalten des Amiga wird nun eine sogenannte Umdenkspannung erzeugt, die der Streustrahlung des Prozessors um 180 Grad Celsius entgegengerichtet ist. Da diese Umdenkspannung gefährlich für den Prozessor ist, benötigen wir eine Kanne mit Mäusemilch, die im Abstand von 2 Minuten 26 Sekunden auf das Plastikgehäuse des Prozessors geträufelt wird (Beim 68020 bitte 1 Minute 32 Sekunden einhalten, bei 68030 lohnt sich das Nachgießen nicht mehr). Die Mäusemilch bekommen Sie, liebe Leser ganz einfach: Lesen Sie unseren Bericht, und in Ihrem Kopf entsteht automatisch der Satz: "Das ist doch zum Mäusemelken!" Tun Sie es doch!!

Durch die Kühlung in Intervallen entwickelt sich eine Hochfrequenz, die sich folgendermaßen berechnet:

$$SP_H = \frac{mm_{Int} * 82 + 2869{,}73}{PI * D}$$

SP_H = Spulenhöhe (m)
mm_{Int} = MäuseMilchIntervall
PI = PI
D = Dau(men)

Die Formel ergibt also eine errechnete Frequenz von 31,31 GigaHertz.

Diese Frequenz wird von der Abschirmplatte des A500 reflektiert und auf den Bildschirm des Monitors geworfen und zwar brutal frontal, weshalb sie auch BruFroTal-Frequenz oder F_{Latsch} (LATente SCHadwelle) genannt wird. Die im Monitor entstehende Bildwiederhol-Frequenz wird mit der Schadwelle gekoppelt und zur Bildisweg-Frequenz, die den vom TV-Satelliten ausgesendeten Signalen täuschend echt gleicht. Der Monitor merkt den Unterschied nicht und gibt ohne Protest jedes Privat-TV-Programm zum besten. Sie können so lange fernsehen, bis Ihnen die Mäusemilch ausgeht oder Ihnen die Seifenopern zum Hals raushängen.

Das AMIGA-DOS-Team wünscht Ihnen viel Glück beim Nachbau.

(A.Pril/jb)

Ralf Zuber

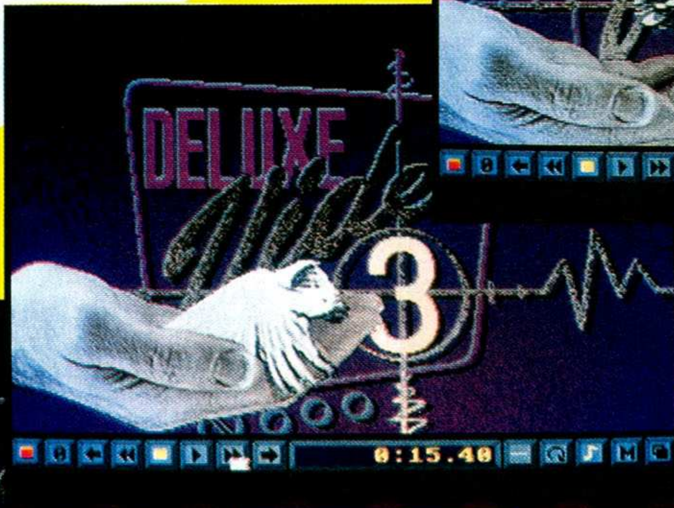
Deluxe Video III

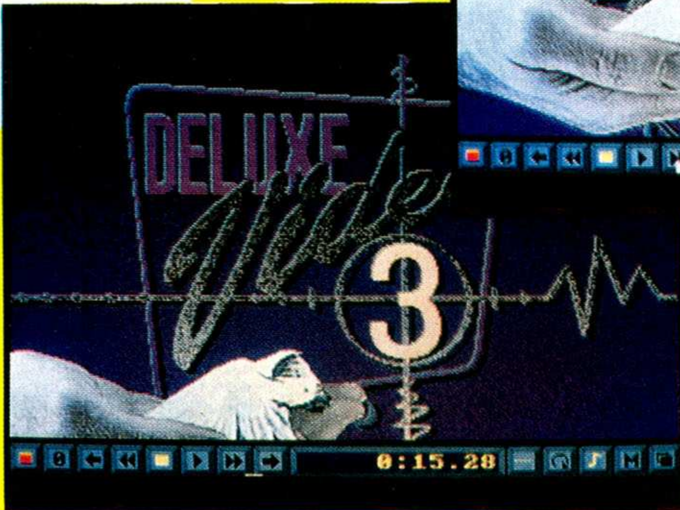
Video mit dem Amiga

Das Raumschiff steht startbereit auf der Rampe. Unter ohrenbetäubendem Lärm zünden die Triebwerke. Der Rauch der Triebwerke verdichtet sich und nimmt den Zuschauern die Sicht. Langsam erhebt sich das Schiff und fliegt dem blauen Himmel und seinem Ziel entgegen.

Nein, wir sind nicht in Cape Canaveral, und wir sind auch keine Zuschauer eines Videofilms. Aber so wie das eben beschriebene Szenario könnte ein Videoclip aussehen, den Sie mit Deluxe Video III, einem Amiga mit genügend Speicher und etwas Phantasie selbst erstellen können.

Ohne bewegte Bilder würde der Amiga nur halb soviel Faszination auf seine Anwender ausüben. Doch von kleinen Bewegungen in Bildern, die mit den meisten Animationsprogrammen realisierbar sind, bis zu einem fertigen Videoclip ist es ein ganzes Stück Arbeit. Deluxe Video III präsentiert sich als Programmpaket, dessen Leistungen weit über einfache Animation hinausreichen. Vielmehr liegt hier ein Produkt vor, mit dem Videos von A bis Z, also in ihrer Gesamtheit aus Bewegung, Musik und Szenenabfolge, erstellt werden können.





Jedoch muß Ihr Amiga einige Voraussetzungen erfüllen, um Deluxe Video III benutzen zu können. Es wird ein Minimum von 1 MByte Speicher und ein zweites Diskettenlaufwerk benötigt.

Sind Sie Besitzer einer Festplatte, können Sie das Programm auch problemlos dort installieren, da auf einen Kopierschutz verzichtet wurde.

Um einer solch komplexen Aufgabe wie der Erstellung eines Videos gerecht zu werden, bedarf es mehr als nur eines einzelnen Programms. Dementsprechend besteht Deluxe Video aus mehreren Programmteilen. Wer ein Video erstellen möchte, muß sich zunächst einmal über den Inhalt des geplanten Streifens klar sein. Mit DV-Maker (Deluxe Video Maker) kann das Drehbuch im Detail vorgeplant werden. Aus diesem Videoscript sind alle Szenen und Effekte ersichtlich. Was das Videoscript für das gesamte Video, ist das Szene-Script für die einzelnen Szenen. Hier muß der Anwender dem Computer mitteilen, was im einzelnen passieren soll. Ein einfaches Beispiel: Die erste Szene eines beliebigen Videoclips soll das digitalisierte Bild einer Frau sein. Nun muß im Szene-Script definiert werden, daß das betreffende Bild, bevor es gezeigt wird, erst einmal von einer Diskette eingelesen werden muß. Ist das Bild im Speicher, kann der Amiga es einfach von einem Sekundenbruchteil auf den anderen einblenden, es langsam aufbauen oder die einzelnen Farben aus einem schwarzen Screen herausschälen. Über die eben beschriebenen Möglichkeiten gibt es noch eine ganze Reihe weiterer Effekte, um Bilder ein- und auszublenden. Auch das Überblenden von einem Bild zum anderen stellt kein Problem dar. Die zur Verfügung stehenden Effekte werden über ein Effektrequester angezeigt und können mit einem Mausklick ausgewählt werden.

Das Programm erlaubt die Benutzung von allen Auflösungen und unterstützt auch den HAM-Modus. Die Größe eines Bildschirms kann bis zu 1024x1024 Punkten betragen. Die Integration von Animationen, die unter Deluxe Paint erstellt wurden, ist problemlos möglich. So richtig Bewegung kommt mit der Benutzung von Brushes auf den Bildschirm, wobei auch hier wieder Animationen und Colorcycling unterstützt werden. Hier offenbart Deluxe Video, wie leicht das Erstellen von Videos sein kann. Wollen Sie zum Beispiel einen Brush von der linken zur rechten Ecke wandern lassen, brauchen Sie nur den Start- und Endpunkt mit der Maus zu be-



stimmen, und den Rest übernimmt das Programm. Doch auch schwierigere Bewegungsabläufe stellen beim DV-Maker kein Problem dar. Nehmen wir mal an, Sie wollen einen im Bogen geworfenen Ball darstellen. Nun müßten Sie normalerweise mehrere Koordinaten der Flugbahn angeben. Nicht so beim DV-Maker. Sie bestimmen wie im ersten Beispiel den Startpunkt und zeichnen dann einfach mit der Maus die Flugbahn, wobei auch die Geschwindigkeit berücksichtigt wird. Das heißt, wenn Sie am Anfang die Maus schneller über den Bildschirm bewegen als am Ende, so wird Ihr Ball im Video das gleiche tun. Doch was ist ein Video ohne Geräusche? Auch hier hilft uns das Programm weiter. Es erlaubt die Verwendung von Sounds, die im 8SVX-Format vorliegen müssen.

Soundtracks – Action auch akustisch

Ein Film ohne Musik ist wie eine Currywurst ohne Ketchup. Auch die Deluxe-Video-Macher haben eingesehen, daß Stummfilme wenig konkurrenzfähig sind. Dementsprechend enthält Deluxe Video III auch Optionen, um Musik und Geräusche in die Videos einzubinden. Diese müssen als IFF-SMUS-File vorliegen.

Auch die Darstellung von Text wird unterstützt. Über einen entsprechenden Requester können Sie unter mehreren verschiedenen einen Zeichensatz auswählen und benutzen, und sogar 'Special Effects' wie Schatten können leicht erreicht werden. DV-Maker erlaubt auch die Verwendung von Genlock und Midi-Instrumenten. Damit dürfte man schon qualitativ hochwertige Videos erstellen können. Nun noch eine kurze Erläuterung zu den anderen Programmen. DV-Player ist ein Programm, das Ihre erstellten Videos abspielt. Das Erfreuliche daran ist, daß dieses Programm Public Domain ist. Somit können Sie Ihre Videos auch an Bekannte schicken und von diesen bewundern lassen. DV-Mover kopiert alle für ein Video benötigten Dateien, zum Beispiel Bilder, Brushes oder Sounds, von den verschiedenen Disketten, so daß sich dann das Video mit allen Dateien zusammen auf einer Diskette befindet. Und zu guter Letzt gibt es noch ein Programm, mit dem man sich eine Dia-Show zusammenstellen kann.

Videos grenzenlos?

Haben Sie Animationen mit DPaint III erstellt, können Sie diese genauso verwenden wie Raytracing-Bilder mit 4096 Farben. Die Möglichkeiten, mit Deluxe Video III seine eigenen Videos zu erstellen, sind mannigfaltig und nach kurzer Einarbeitungszeit leicht zu bewältigen.

Neben einer logischen und übersichtlichen Benutzerführung besticht das Programm mit einer, wie bei Electronic-Arts-Produkten üblich, ausgezeichneten Dokumentation. Das Buch enthält neben detaillierten Erklärungen aller Programmteile auch ein Tutorial, in dessen Rahmen der Benutzer anhand klarer Beispiele in die Deluxe-Video-Benutzung eingeführt wird. Programme und Handbuch sind komplett in Englisch und damit das einzige, echte Manko, das der deutsche Benutzer hier in Kauf nehmen muß.

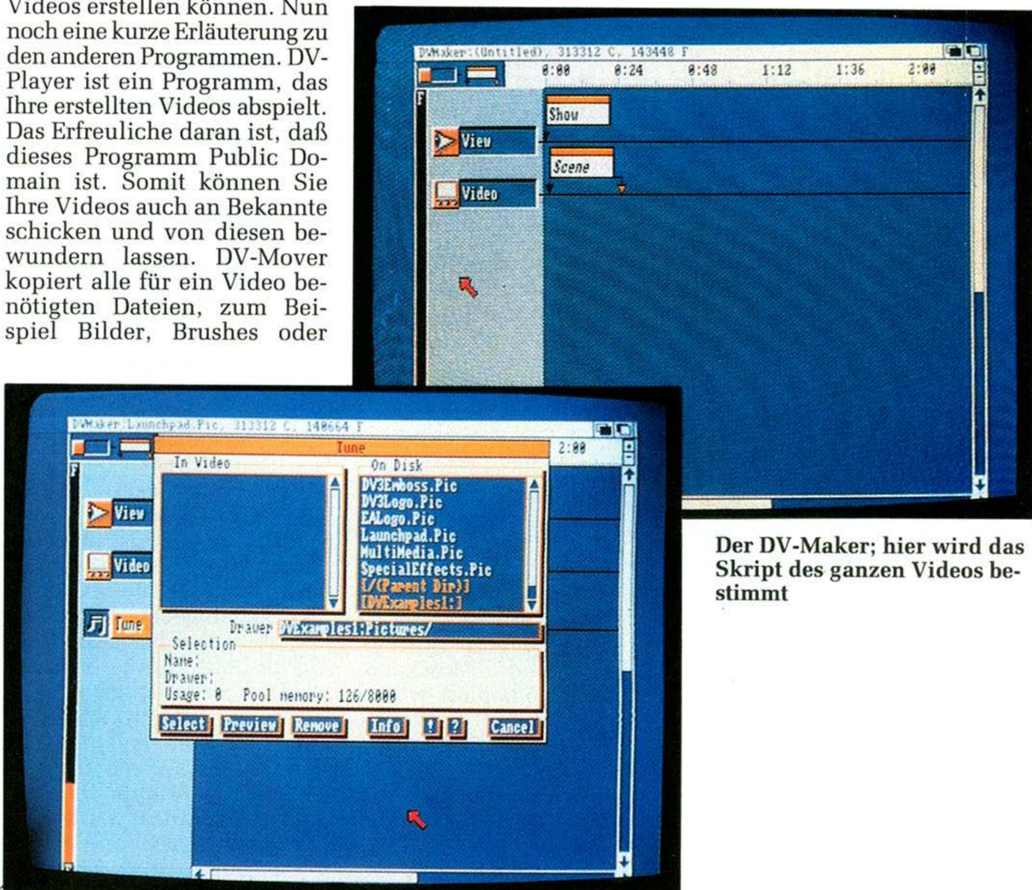
(R. Zuber/hs)

AMIGA DOS
Blitzlicht

Name: Deluxe Video III
Hersteller: Electronic Arts
Vertrieb: Fachhandel
Preis: voraussichtlich zirka 300 DM
Besonderheiten: mindestens 1 MByte Speicher und zweites Laufwerk erforderlich

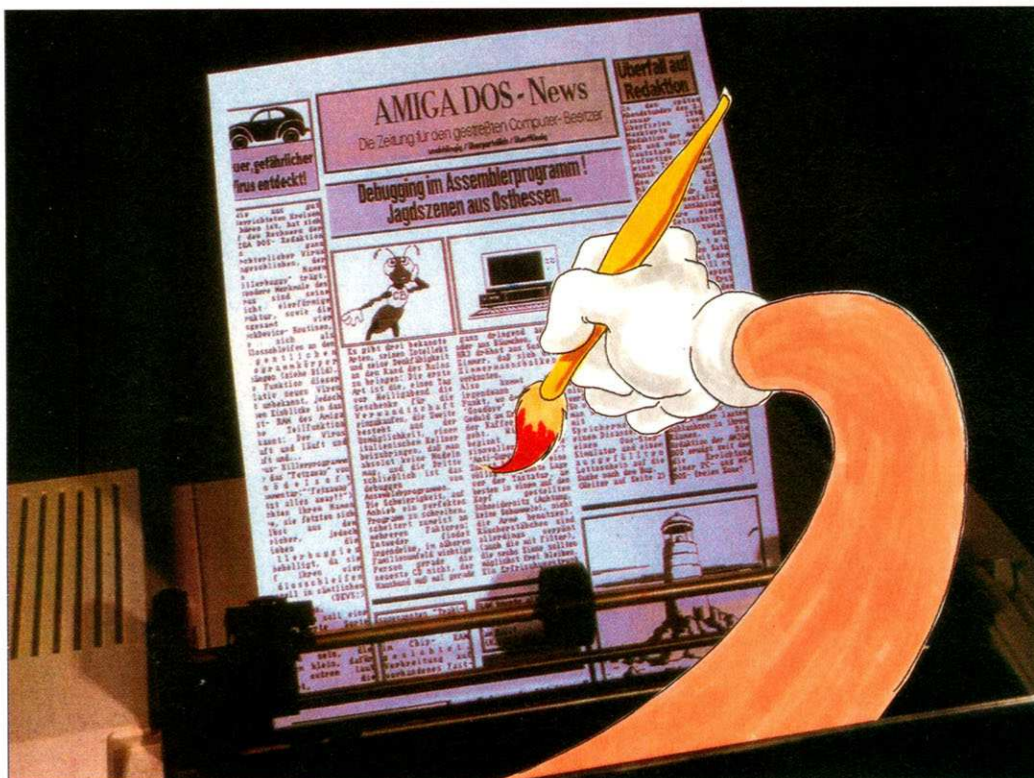
Positiv:
- klare Benutzerführung
- leicht erlernbar

Negativ:
- Handbuch und Programm komplett in Englisch



Der DV-Maker; hier wird das Skript des ganzen Videos bestimmt

Mit Preview können Videos, die in der Entwicklung sind, unter die Lupe genommen werden



Andreas Polk

Schönschrift für den Matrixdrucker

Der Postscript-Interpreter für den Amiga

Desktop Publishing ist zwar inzwischen auch für den Heimbereich attraktiv geworden (verschiedene Programme beweisen das), jedoch wird die Ausdruckqualität durch den Umstand gemindert, daß an den meisten 'Heimcomputern' (zu denen inzwischen auch der Amiga und PC gehören) ein Matrixdrucker seine Dienste verrichtet.

Dies liegt jedoch weniger an dem Umstand, daß die Programme nicht gut sind oder DTP nicht für den Heimbereich zu gebrauchen ist, sondern daran, daß die Druckdateien für die entsprechenden Seiten im PostScript-Format vorhanden sind. PostScript ist eine spezielle Sprache für High-Tech-DTP-Anlagen wie die hier im Heft vorgestellte. Matrixdrucker können mit PostScript-Dateien soviel anfangen wie der Eskimo mit einem Kühlschrank (jedenfalls solange er noch in den Kälteregionen der Erde lebt). Was braucht also der DTP-Begeisterte, der einen vernünftigen Ausdruck auf seinem Matrixdrucker haben will? Richtig, einen PostScript-Interpreter. Und damit wären wir beim Thema.

Die einzigen Drucker, die die Sprache PostScript im Augenblick verstehen, sind die entsprechenden Laserdrucker. Mit PixelScript soll sich dies nun ändern, denn es können alle Postscript-Dateien auf den gängigen Preferences-Druckern ausgedruckt werden.

Wie sag ich's meinem Drucker?

Zuerst einmal zu der Frage, was PostScript überhaupt ist. PostScript ist eine Sprache, die nur für die Druckeransteuerung konzipiert wurde. Der Drucker wird nicht mehr über Kontrollsequenzen wie beispielsweise den ESC/P-Standard angesprochen, sondern über eine eigene Programmiersprache. Der Unterschied zu anderen Programmiersprachen besteht darin, daß PostScript im Gegensatz zu C oder einer anderen Sprache nur für die Ausgabe von Daten auf den Drucker zuständig ist. Somit besitzt diese Sprache sehr viele nützliche Befehle, und der Ausdruck ist entsprechend gut. Der Nachteil an der ganzen Angelegenheit ist jedoch der, daß nur sehr teure Drucker – in der Regel Laserdrucker ab 10000 DM – diese Sprache verstehen, denn dazu ist ein PostScript-Interpreter notwendig. Dies ist eine Hardware, die im Drucker eingebaut ist und nur die Aufgabe hat, die PostScript-Datei zu interpretieren und auszudrucken. Diese Hardware ist recht teuer, dafür aber auch entsprechend schnell. Es können mehrere Seiten pro Minute interpretiert und gedruckt werden. Da aber kaum jemand über einen solch teuren Drucker verfügt (wer hat schon 10000 bis 30000 Mark Kleingeld in der Brieftasche), muß hier Abhilfe geschaffen werden. Die preiswerteste Methode besteht darin, den Hardware-Interpreter durch Software zu ersetzen. Somit können nicht nur Laserdrucker, sondern jeder beliebige Preferences-Drucker angesprochen werden.

PixelScript soll die Lösung heißen

PixelScript ist das erste Programm für den Amiga, das diese Aufgabe bewältigt. Es wandelt also PostScript-Dateien so um, daß sie auf jedem Drucker ausgedruckt werden können, für den ein Preferences-Treiber existiert. Natürlich müssen hierbei Abstriche gemacht werden. Zum einen dauert das Interpretieren einer Seite – also das Umwandeln der PostScript-Datei in eine für den Drucker verständliche Form – viel länger als bei PostScript-fähigen Druckern. Wartezeiten von einer Stunde sind bei komplexen Seiten nicht selten. Außerdem kann der Ausdruck eines 24-Nadel-Druckers auch in der höchsten Auflösung nicht mit dem Ausdruck eines Laserdruckers mithalten.

So, nun aber genug der Einführung. Widmen wir uns dem Programm PixelSript einschließlich seiner Vor- und Nachteile. Zuerst muß PixelScript installiert werden. Hierzu wird ein Installationsprogramm mitgeliefert, das die Installation ausreichend vornimmt. Der einzige Nachteil ist der, daß das Programm DPS, auf das gleich noch eingegangen wird, nicht automatisch mit installiert werden kann. Hier muß man über das CLI gehen. Startet man PixelScript von der Workbench aus, so wird ein neuer Screen geöffnet. In der linken oberen Ecke befindet sich ein Rad, welches anzeigt, ob PixelScript gerade rechnet oder nicht. Nach kurzer Zeit kann der Benutzer Befehle eingeben.

Bevor man nun erwartungsvoll den ersten Ausdruck startet, sollte man erst einmal schauen, ob PixelScript auch richtig konfiguriert, also an die vorhandene Hardware angepaßt ist. Hierzu wählt man den Menüpunkt 'Configure' aus. Es erscheint ein neues Fenster mit vielen Einstellmöglichkeiten. Neben den Einstellungen, die schon durch Preferences festgelegt sind, sind noch einige zusätzliche möglich.

PixelScript

Abb. 1. Probeausdruck 360*360 dpi mit Excellence!...

PixelScript

Abb. 2. ...und mit PixelScript (Originalgröße)

So kann beispielsweise festgelegt werden, daß der Druckbereich des Blattes kleiner ist als in Preferences angegeben. Auch ein optionaler Treiber kann bestimmt werden. Besitzen Sie nämlich einen nicht PostScript-fähigen Laserdrucker, so brauchen Sie nicht den Umweg über den (nicht vorhandenen) Preferences-Treiber zu gehen, sondern können hier einen Treiber auswählen.

Die Konfiguration des PostScript-Umwandlers: Aller Anfang ist schwer

Für diesen Treiber können dann noch andere Angaben gemacht werden, wie beispielsweise die Festlegung der Druckauflösung. Möchten Sie IFF-Dateien drucken, so können Sie in dem Gadget Screen-Frequency noch zusätzliche Einstellungen vornehmen. Auch der Rasterwinkel einer Grafik, die sich aus Graustufen zusammensetzt, ist einstellbar.

Als Besonderheit bietet PixelScript ferner die Möglichkeit, eine Datei nicht auf den Drucker auszudrucken, sondern in eine Datei zu schreiben. Dies ist vor allem dann sinnvoll, wenn Sie sich eine PostScript-Datei erst einmal anschauen wollen, bevor Sie den Druck starten. Um eine so entstandene Datei anzuschauen, befindet sich das Public-Domain-Programm DPS im Lieferumfang. So lassen sich mögliche Fehler leichter auffinden, da nicht erst ein Ausdruck, der wesentlich länger dauert als das 'Drucken' in eine Datei, gestartet werden muß. Außerdem können Sie von den entstandenen Grafiken Hardcopies anfertigen und in Präsentationsprogrammen weiterverarbeiten.

Möchten Sie eine bereits in eine Datei geschriebene Datei nachträglich ausdrucken lassen, so brauchen Sie die PostScript-Datei nicht noch einmal durch den Interpreter zu schicken. Das Programm PxSPrint übernimmt diese Aufgabe. Allerdings druckt es nicht in der höchsten Auflösung, sondern in der, die beim Druck in eine Datei angegeben wurde. Da dies meistens 72*72 dpi ist, wird man in den meisten Fällen um ein nochmaliges Interpretieren nicht herumkommen, möchte man das beste Ergebnis erzielen. (Das Programm PxSPrint darf übrigens frei kopiert werden.)

Haben Sie alle Einstellungen vorgenommen, so können Sie diese unter einem speziellen Namen abspeichern. Auch das automatische Laden Ihrer Konfigurationsdatei beim Start von PixelScript ist möglich. Haben Sie die Konfiguration eingestellt, so müssen Sie nur noch die zu druckende PostScript-Datei auswählen. Hierzu gibt es eine File-Select-Box. Nach Auswählen einer Datei beginnt PixelScript mit dem Interpretieren der Datei und druckt sie dann aus. Dieser Vorgang kann bei ganz einfachen Seiten recht schnell gehen, bei komplexen Seiten können aber Wartezeiten von zirka einer Stunde auftreten.

Ein starkes Gespann gegen PostScript-unwillige Drucker

Dafür läßt sich das Ergebnis sehen. Mit einem 24-Nadel-Drucker in der höchsten Auflösung von 360*360 dpi gleicht das Ergebnis schon stark dem Ausdruck eines Laserdruckers. Jegliche Rundungen der Buchstaben erscheinen auch tatsächlich rund. Der Treppchen-Effekt entfällt fast völlig. Nur bei genauem Hinschauen sind einzelne Punkte, die allerdings durch die Größe der Nadeln bedingt sind, erkennbar. Da PixelScript die Druckertreiber der Workbench benutzt (es sei denn, Sie besitzen einen Laserdrucker und nehmen die PixelScript-eigenen Treiber), ist es möglich, das Programm TurboPrintII, welches die Workbench-Treiber durch eigene, bessere ersetzt, zu nutzen. Somit kann der Ausdruck noch einmal beschleunigt werden. Hier ist das Programm TurboPrintII eine zu empfehlende Ergänzung zu PixelScript.

Sollte einmal ein Fehler aufgetreten sein, so erscheint ein Requester, der die Fehlerart anzeigt. Der geübte Anwender kann hier nun Korrekturen vornehmen und den Aus-

Einige Tips zum Arbeiten mit PixelScript

An dieser Stelle sollen Ihnen vier kleine Tips gegeben werden, die vielleicht helfen, wenn PixelScript mal nicht so arbeitet, wie Sie es möchten: – Sollten unerklärliche Fehlermeldungen auftreten, installieren Sie PixelScript einfach noch einmal. Bei uns half diese Prozedur.

– Bedenken Sie, daß PixelScript die Einstellungen von Preferences benutzt. Ein guter Wert für die Papiergröße ist 8,2 für width und 11,5 für height mit aktiviertem Bounded-Gadget.

– Bedienen Sie PixelScript über das CLI, so müssen Sie die Befehle angeben, wie PixelScript sie im Command-Gadget der Intuition-Oberfläche schreibt. Um einen Ausdruck zu starten, geben Sie beispielsweise *(PXS:PXSFiles/Test.ps) run* ein. Dies ist auch der Befehl, den PixelScript im Command-Gadget ausgibt, wenn Sie mittels Select-File die Datei PXS:PXSFiles/Test.ps angegeben haben.

– Erscheint immer ein Error, so geben Sie einfach 'Abort' oder 'Resume' an, das normale PixelScript erscheint dann wieder.

ᴄken Sie PostScript-Dateien auf Ihrem
ferences-Drucker.

PixelScript

Versio
Der PostScript-Interpreter für den Am

Schriftpro
Helvetica 6pt
Helvetica 8pt
Helvetica 10p
Helvetica 12
Helvetica 1
Helvetica 1

DTP

Textprobe

Einleitung
PixelScript ist der PostScript-Interpreter für den Amiga Computer. Er ermöglicht es Ihnen, PostScript-Dateien auf Ihrem Matrix-, Tintenstrahl- oder Laserdrucker in der höchsten Auflösung auszugeben. Solche PostScript-Dateien werden von vielen DTP- und Textprogrammen erzeugt.
Mit PixelScript können Sie eine Druckqualität erreichen, die sonst nur auf PostScript-Druckern möglich ist. PixelScript ist ein leistungsfähiger Interpreter, der die meisten Fähigkeiten von PostScript



Kurfürstendamm 64-65
1000 Berlin 15
Tel.: (030) 88 33 505

Seite wurde mit PageStream gesetzt, als PostScript-Datei ausgege
mit PixelScript gedruckt.

Abb. 3. Die Demo-Seite wurde mit PixelScript auf einem 24-Nadel-Drucker ausgedruckt: hier die Verkleinerung...

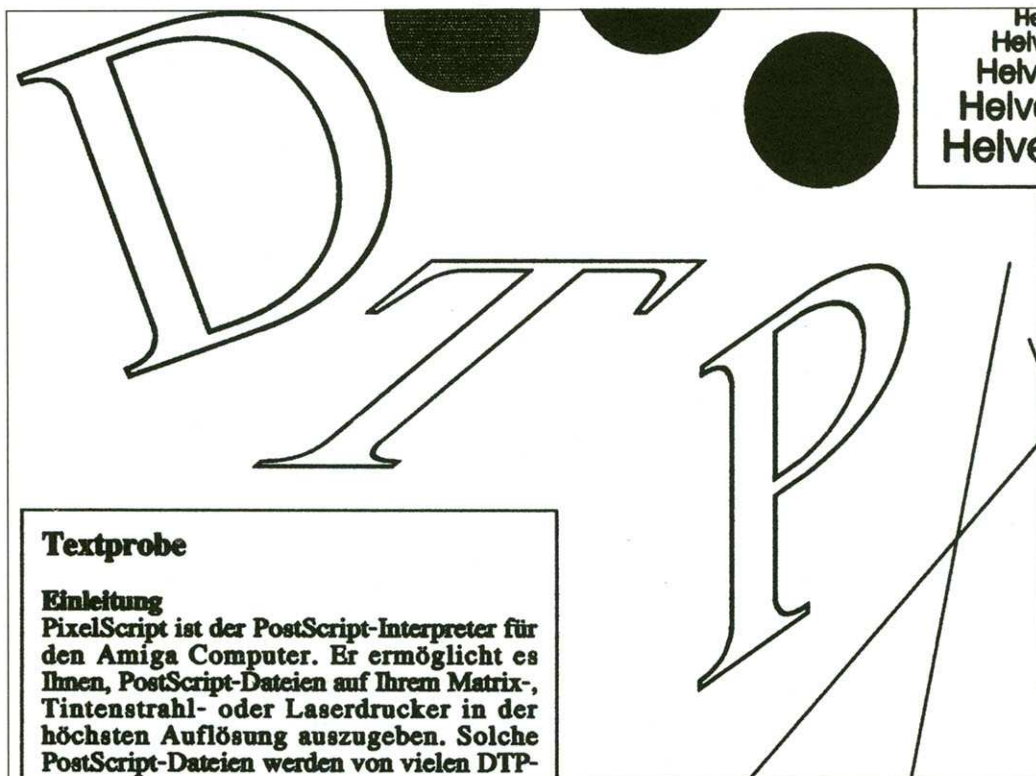


Abb. 4. ...und hier ein Ausschnitt in Originalgröße

druck wiederholen oder einfach den Fehler übergehen. Wenn Sie die Sprache PostScript lernen möchten, ist PixelScript auch hier eine nützliche Hilfe. In einem Command-Gadget können einzelne PostScript-Befehle eingegeben werden. Diese werden dann direkt interpretiert und ausgedruckt. So fällt das Lernen der Sprache wesentlich einfacher, da Befehle ausprobiert und die Ergebnisse direkt angeschaut werden können.

Nun fragt man sich natürlich, wo solche PostScript-Dateien überhaupt herkommen. Man kann sie natürlich selber programmieren. Dies ist jedoch sehr kompliziert und umständlich. Viele Programme bieten die Möglichkeit, PostScript-Dateien zu erzeugen. Dazu gehören die meisten DTP-Programme, wie beispielsweise PageStream oder Professional Page. Auch einige Textverarbeitungen bieten diese Möglichkeit, wie beispielsweise 'Excellence!'.

Beim Test benutzte ich PostScript-Dateien von PageStream und von Excellence!. Hierbei gab es keine Probleme. PixelScript hat diese Dateien alle akzeptiert und gedruckt. Die Qualität des Ausdrucks mit PixelScript ist wesentlich höher als die der Ausdrucke der Programme.

Natürlich hat auch PixelScript einige Nachteile, die hier nicht unterschlagen werden sollen. Zuerst wäre hier der überaus hohe Speicherbedarf anzuführen. Mit 512 kByte kommt man nicht aus, 1 MByte ist das Minimum, reicht aber auch nicht immer aus. Eine 2-MByte-Erweiterung ist zu empfehlen. Möchten Sie Speicherplatz sparen, so bietet PixelScript die Möglichkeit, die Intuition-Oberfläche zu unterbinden. So werden laut Handbuch 170 kByte gespart. PixelScript wird dann interaktiv über das CLI gesteuert.

Hier wären wir auch schon beim nächsten Minuspunkt: das Handbuch. Es erwähnt zwar, daß man PixelScript interaktiv über das CLI bedienen kann, aber wie wird nicht verraten. Der Benutzer muß die Befehle selbst herausfinden. Von einem Programm, welches 300 DM kostet, kann man schon ein komplettes und ausführliches Handbuch erwarten. Zuletzt ist eigentlich nur noch der Preis zu nennen. 298,– DM für ein Utility, denn das ist PixelScript ja letztendlich, ist recht viel. Hier muß der Anwender selber entscheiden, ob er diese Summe investieren möchte. Zuerst muß man sich im klaren sein, daß PixelScript seine hauptsächliche Anwendung im Desktop-Publishing-Bereich findet. Damit ist ein DTP-Programm (oder eine Textverarbeitung, die in der Lage ist, PostScript-Dateien zu erzeugen) schon fast ein Muß. Ist man dabei jetzt auf guten Ausdruck angewiesen, wird man diese Summe sicherlich zahlen.

Die Testergebnisse zusammengefaßt

Summiert man alle Vor- und Nachteile, so kommt man zu dem Schluß, daß PixelScript ein gutes Programm ist, welches seinen Zweck sehr gut erfüllt.

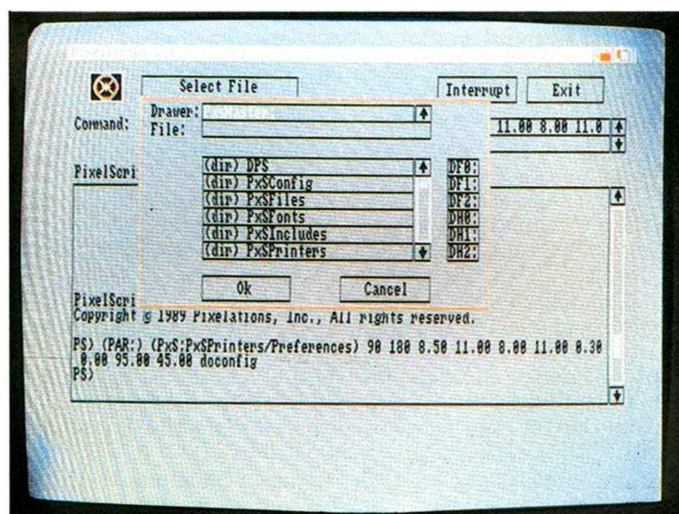
Die Druckergebnisse sind auch mit einem Nadeldrucker sehr gut. Die Geschwindigkeit läßt sich durch gleichzeitiges Verwenden von TurboPrintII noch beschleunigen. PixelScript zielt allerdings mehr auf den gehobenen Anwendungsbereich, da es sehr viel Speicher benötigt und auch nicht gerade billig ist. Der vorhandene ARexx-Port läßt hoffen, daß jemand einmal eine Schnittstelle zu einem Anwenderprogramm definiert. Somit könnten Ausdrucke direkt von Anwenderprogrammen aus gestartet werden! (jb)

AMIGA DOS Blitzlicht

Name: PixelScript
Anbieter: Gold Vision, Kurfürstendamm 64-65, 1000 Berlin 15, Tel.:030/8833505
System: Alle Amiga ab 1 MByte (je mehr Speicher, desto besser)
Besonderheiten: Drucken in Datei, eigene Druckertreiber für Laserdrucker, zusätzliche Utilities

Positiv:
- sehr guter Ausdruck
- Eingabe einzelner PostScript-Befehle möglich
- Konfiguration abspeicherbar
- Starten ohne Intuition möglich (Speicher sparen)
- Unterstützung von ARexx

Negativ:
- schlechtes Handbuch
- viel Speicher nötig
- hoher Preis

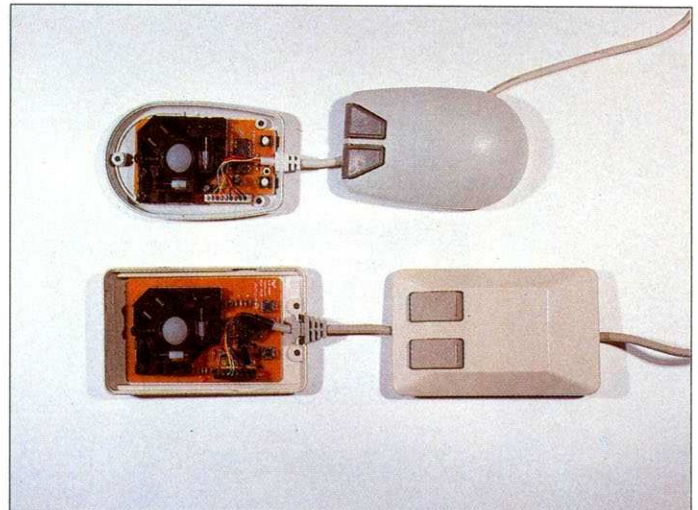


Der 'Configure-Screen' von Pixelscript

Mäusemächtige Eingabemedien

Reisware- und Amiga-Maus im Wettstreit

Wer den Amiga gerne unter Intuition benutzt, kommt ohne Maus nicht aus. Bisher waren Amiga-Anwender bei einem Versagen des rollenden Eingabemediums auf original Commodore-Ersatzteile angewiesen. Mit der Reisware-Maus gibt es nun eine echte Alternative.



Beiden Mäusen in den Bauch geschaut. Das Innenleben beider Geräte unterscheidet sich nur unwesentlich voneinander

Oval, in kargem Grau gehalten, macht die Reisware-Maus auf den ersten Blick einen schlichten Eindruck. Jedoch braucht sich die Reisware-Maus auch nicht hinter Äußerlichkeiten verstecken. Die ovale Form ist sehr grifffreundlich und der Anatomie der Handinnenflächen angepaßt. Die Maustasten sind, wie auch bei der original Commodore-Maus, an der Vorderseite angebracht.

Was das Innenleben anbelangt, unterscheiden sich die beiden Mäuse nur unwesentlich voneinander. Beide weisen die obligatorischen Mikroschalter als Maustasten auf (ältere Amiga-Mäuse sind teilweise noch nicht mit Mikroschaltern ausgerüstet). Für das restliche Innenleben zeichnet sich bei beiden Mäusen Mitsumi Elec. verantwortlich.

Tatsächlich unterscheiden sich beide Mäuse im wesentlichen durch das Design, das bei der Reisware-Maus anwenderfreundlicher gestaltet wurde.

Die Reisware-Maus kostet 89, – DM und genießt einen zwölfmonatigen Garantieschutz.

Die Maus von Commodore kostet 99, – DM und hat sechs Monate Garantie.

(hs)



Desktop Publishing professionell

Das Desktop-Publishing-Paket in der AMIGA-DOS-Redaktion

Ein 'neuer' Amiga in der AMIGA-DOS-Redaktion? Des Rätsels Lösung: Hier wird ein Amiga zum professionellen Arbeitsgerät. Die mitgelieferte Peripherie bringt dies noch besser zur Geltung – Postscript-Laser-Drucker, eine 68020-Karte und 3 MByte RAM – das alles gehört zum neuen Betätigungsfeld des Amiga-Desktop-Publishing.

Wer die DTP-Anlage geliefert bekommt, steht zunächst vor einem Platz-Problem: Der Laserdrucker von NEC, ein 'Silentwriter LC890' braucht davon eine ganze Menge. Zusammen mit Amiga und Monitor wird der Platz recht schnell knapp, wenn man nicht von vornherein den Platzbedarf einkalkuliert.

Für wen soll nun diese Anlage sein? Heimanwender werden wohl kaum zu den Abnehmern gehören, es sei denn, man gehört zur schweigenden Mehrheit derjenigen, die mal eben locker ca. 30000 DM mit sich herumtragen.

Für die Heimarbeit ist die DTP-Anlage wirklich nicht geschaffen. Anders dagegen im Business: Werbeleute, Agenturen, die Präsentationen erstellen, dies sind die potentiellen Käufer einer solchen DTP-Anlage.
Und für deren Geldbeutel gibt die Anlage schon einiges her.

Ein Rennpferd mit Starallüren – der 68020

Wichtigste Verbesserung zu den 'herkömmlichen' Amigas ist der Einbau eines 68020-Prozessors anstelle des sonst üblichen 68000 in einen Amiga 2500. Die Vorteile des Prozessors sind schnell aufgezählt: Volle 32 Bit Rechenleistung liegen hier vor. Bemerkbar macht sich dies vor allem beim Berechnen der PostScript-Dateien. Die mitgelieferte 68020-Karte enthält außerdem einen 68881-Coprozessor, der den Hauptprozessor bei arithmetischen Operationen noch weiter entlastet.

Der Amiga 2500 enthält ebenfalls schon den neuen Big-Agnus-Chip. 1 MByte direkt adressierbares Chip-RAM ist also direkt vorgesehen. Beim Monitor wird auch nicht gespart; ein hochauflösender Multisync-Monitor übernimmt die Rolle, die bisher dem sonst üblichen 1084 vorbehalten war. Aus gutem Grund, wie wir nachher noch sehen werden. Schließlich der Laserdrucker; er ist wohl das Aushängeschild der DTP-Anlage. Hier sind vollständig PostScript-fähige Ausdrucke möglich , die auf 'normalen' Matrix-Druckern noch immer Zukunftsmusik sind und auch bleiben werden. Dabei bietet der 'Silentwriter LC890' noch mehr Komfort. Sollte er einmal nicht als PostScript-Drucker benötigt werden, so kann er auf eine Hewlett-Packard-Laserjet-Emulation umgeschaltet werden. Diese Emulation wird sogar von den Preferences-Druckertreibern unterstützt. Somit sind ganz einfache Ausdrucke unter der Shell oder den entsprechenden Anwendungsprogrammen möglich. Die Voreinstellungen des Druckers werden an diesem direkt vorgenommen, ein Menue führt dabei durch sämtliche Funktionen. Für unseren Test beließen wir es bei der parallelen Schnittstelle als Datenempfänger und der PostScript-Bearbeitung.

PostScript – Mein Drucker versteht mich besser

PostScript hört sich im ersten Augenblick sehr exotisch an, ist jedoch recht einfach zu erklären: Um Drucker direkt anzusteuern, gibt es die Möglichkeit, die Steuercodes per ESC-Code oder per PostScript zu übermitteln. PostScript ist eine spezielle Computersprache, die nur für Drucker interessant ist. Denn alles was in PostScript geschrieben wird, dient nur der Druckerausgabe. PostScript-Dateien müssen allerdings in ein spezielles Format umgewandelt werden, welches der Drucker als seine Steuerzeichen versteht. Diese Umwandlung geschieht über spezielle PostScript-Interpreter, wobei eine nahe Verwandtschaft zu anderen, bekannteren Computersprachen wieder durch das Wort Interpreter gegeben ist. Diese Interpreter wandeln das von uns geschriebene Druckprogramm in für den Drucker verständliche Werte um.

Warum aber PostScript? Nun, der erste positive Punkt ist der, daß man sich nicht mehr mit den Drucker-Codes herummühen muß, gleichzeitig ist damit eine Anpassung an andere Drucker wesentlich einfacher. Auch unsere Zeitschrift entsteht zum Beispiel

Besuchen Sie uns auf der Hobbytronic in Dortmund · Besuchen Sie uns auf der Hobbytronic in Dortmund

- Entwicklung
- Produktion
- Hardware
- Software
- Service



Ihr Amiga-Spezialist

AMIGA-Bremse *"der Highscore-Killer"* DM 39,50
- regelt die Geschwindigkeit stufenlos bis zum Stillstand
- ideal für schnelle Games und Bildschirmfotografie

Neu Neu Neu Neu Neu Neu Neu Neu Neu Neu Neu Neu
Amiga-Bremse für A500 extern mit LEDs DM 69,–
Neu Neu Neu Neu Neu Neu Neu Neu Neu Neu Neu Neu

!Achtung! Preisänderung bei Festplatten !Achtung!

HK-COMPUTER Festplatten

Kapazität	Speed	Filecard	A2000	A500
20MB/31/2"	35ms	898,–	848,–	1048,–
30MB/51/4"	65ms	–	898,–	1098,–
30MB/31/2"	35ms	1048,–	998,–	1198,–
40MB/51/4"	28ms	–	1198,–	1398,–
50MB/31/2"	35ms	1298,–	1248,–	1448,–
60MB/51/4"	28ms	–	1298,–	1498,–

Alle unsere Festplatten werden wahlweise mit ALF V1.6 oder mit Autoboot-Modul ausgeliefert.

AUTOBOOTMODUL für A2000 DM 119,–
AUTOBOOTMODUL für A500 DM 149,–
zum Nachrüsten, inkl. Software und Anleitung
(bitte Harddisk- und Controllertyp angeben)

Festplatten-Controller
OMTI 5520B für MFM-Platten (20/40MB) DM 149,–
OMTI 5528B für RLL-Platten (30/50/60MB) DM 189,–

Festplatten-Interface DM 99,–
Die Adapterplatine paßt den PC-BUS eines Festplatten-Controllers an den AMIGA-BUS an.
(Bitte Rechnertyp angeben)

ALF-Festplattentreiber Version 1.6 DM 98,–
ALF-Festplattentreiber Version 2.0 DM 198,–

Autoboot-Set MFM
OMTI 5520B, Autoboot-Modul, Festplatten-Interface, Kabelsatz
für Amiga 2000 (interne Slotkarte) DM 349,–
für Amiga 500 (extern mit Gehäuse) DM 379,–

Autoboot-Set RLL
OMTI 5528B, Autoboot-Modul, Festplatten-Interface, Kabelsatz
für Amiga 2000 (interne Slotkarte) DM 389,–
für Amiga 500 (extern mit Gehäuse) DM 419,–

Festplattengehäuse A500/1000 DM 379,–
Amigafarbenes Metallgehäuse, komplett mit Schaltnetzteil, Lüfter, Anschlußadapter mit durchgeführtem Bus, LEDs

Autoboot-Filecard DM 369,–
Harddisk-Trägerplatine mit integriertem Autoboot-Modul, Controlleradapter, Autoboot-Software

Disketten

31/2" NoName 2DD, 10St. DM 14,95
31/2" Verbatim Verex 2DD, 10St. DM 25,–
31/2" Sentinel 2DD ColorDisk, 10St. DM 29,50
lieferbar in weiß, gelb, orange, rot und grün
51/4" NoName 2S2D, 10St. DM 5,90
51/4" Verbatim Verex, 10St. DM 14,90
Staffelpreise auf Anfrage

Kick-ROM DM 49,–
- Kickstartumschaltplatine für zwei OriginalROMs

Kick-ROM mit ROM 1.3 DM 98,–
OriginalROM 1.3 DM 65,–

Kickstartumschaltplatine 3fach DM 59,–
- für zwei OriginalROMs und eine Epromversion

Umschaltplatine mit einem Epromsatz DM 155,–
Umschaltplatine mit OriginalROM1.3 DM 108,–

Maus & Joystick-Adapter DM 44,50
- für gleichzeitigen Anschluß von Maus und Joystick
- mit LED-Anzeige
- alle Maussteuerleitungen elektronisch geschaltet

»PowerFire« *Das Superding!* DM 29,50
- Dauerfeuerinterface für Joystick und Maus
- optimale Impulsfolge für jedes Game einstellbar
- Dauerfeuer wird über Feuer- bzw. Maustaste aktiviert
- einfach zwischen Maus/Joystick und Rechner stecken
- abschaltbar

Trackdisplay A2000 intern DF0: & DF1: DM 98,–
(neuer Power-LED Träger mit integrierten 7-Segmentanzeigen)

Trackdisplay extern DF0: bis DF3: DM 79,–
für jedes Laufwerk einstellbar

Drive-Expander DM 39,–
- für externe Laufwerke ohne Busdurchführung
- einstellbare Laufwerksnummer
- keine Kabellängenprobleme
- abschaltbar
- bei Verwendung eines Boot-Selectors kann von jedem Laufwerk gebootet werden

BOOT-Selector für Amigas DM 14,50
wahlweise Booten von DF0: oder DF1: oder DF2: oder DF3:
bei Bestellung bitte angeben

BOOT-Selector elektronisch DM 49,–
wahlweise Booten von DF0: bis DF3:

BTX/VTX Decoder mit FTZ (Drews) DM 248,–
Commodore BTX mit FTZ DM 179,–

Midi-Interface DM 89,–
In/Thru/2*Out, im Metallgehäuse für A500/A2000

Staubschutzhauben
AMIGA 500 DM 16,50
AMIGA 2000 Keyboard DM 16,50
Monitor 14" DM 29,50
Drucker 10" DM 24,50
Drucker 15" DM 29,50
Die Staubschutzhauben sind aus Kunstleder mit weichem antistatischem Innenfutter

Software

RAM-Test Amiga DM 24,50
zeigt defekte Speicherstellen grafisch an, 100% Assembler
XCOPY II DM 49,–
XCOPY II mit Hardwarezusatz DM 69,–
Turboprint II DM 89,–
Turboprint professional DM 189,–
Quarterback (Festplatten-Backup) DM 119,–
DPaint III DM 248,–
Beckertext DM 189,–

AMIGA-Computer
Amiga 2000 DM 1898,–
Amiga 500 DM 928,–
Amiga 500 BTX DM 998,–
Harddisk A590 20MB für A500 DM 998,–
Colormonitor Commodore 1084P DM 598,–

Professional Drive Diskettenlaufwerke
**** Top in Qualität, Funktion und Design ****
3 1/2"-Laufwerk AMIGA 2000 intern DM 159,–
komplett mit Einbaukit und Anleitung
3 1/2"-Laufwerk für alle AMIGAs extern DM 229,–
abschaltbar, Busdurchführung, AMIGAfarben
51/4"-Laufwerk für alle AMIGAs extern DM 279,–
abschaltbar, Busdurchführung, 40/80 Tracks, AMIGAfarben

Professional RAM-Board II A500 auf 1 MB DM 199,–
- superschnelle Megabit-RAMs (4*511000)
- mit Uhr & Datum
- hard- und softwaremäßig abschaltbar
- Superniedriger Stromverbrauch

dto. Platine mit Uhr & Schalter ohne RAMs DM 79,–

Professional RAM-Board III A500 auf 2,3 MB DM 698,–
- intern, inkl. Garry-Adapter
- superschnelle Megabit-RAMs (16*511000)
- mit Uhr & Datum
- Hard- und Softwaremäßig abschaltbar

dto. Platine mit Uhr & Schalter ohne RAMs DM 198,–

Professional RAM-Board A2000 DM 848,–
8 MB mit 2 MB bestückt, autokonfigurierend, einfach durch Umstecken zusätzlicher RAMs und Jumper aufrüstbar, keine neuen PALs erforderlich
dto. Platine teilbestückt ohne RAMs DM 498,–
RAM-Satz für 2 Megabyte DM 498,–

Genlock-Interface DM 248,–
nur Computer/nur Video/beide Bilder gemischt
durchgeführter Monitorport, externes Gehäuse
mit S-VHS und Cinch Buchsen

Professional View

Video-Digitizer der neuen Generation DM 998,–
On Board RGB-Splitter bereits integriert !!
Genlock-Interface im Lieferumfang enthalten !!
Realtime-Digitalisierung in 16 Graustufen, 25 Bilder/s
HAM-Modus mit 4096 Farben,
kurze Realtime Animationen in s/w, Übertragung 3,5MB/s !
Alle Bilder und Animationen auf Diskette speicherbar,
für Animationen mind. 1MB RAM
unterstützt 1MB ChipRAM;
Demo gegen frankierten Rückumschlag (DM 2,40) mit LeerDisk.

Haben Sie Hard- oder Software für den Amiga entwickelt ?
Wir bieten Ihnen eine großzügige Umsatzprovision und eine ehrliche Abrechnung.
> > > > > Sprechen Sie uns an < < < < <

Wir reparieren Ihren Amiga und Zubehör schnell und preisgünstig.

UPS-Versand: Nachnahme + 10 DM, Vorauskasse + 5 DM, Großgeräte nach UPS-Tabelle ohne Aufschlag, Ausland nur gegen Vorauskasse + 15 DM.
Bei Vorauskasse nur Eurochecks bis 400,– DM oder Überweisung.
Fordern Sie unser kostenloses Info an.
Händleranfragen erwünscht!

HK-Computer

F.Hansmann & Th.Küpper GbR
Bonner Straße 37 * 5000 Köln 1

Telefon: 0221/311606 ** Telefax: 0221/321166
Mo-Fr. 10.00-13.30 u. 14.30-18.30, Sa. 10.00-14.00
BTX: 0221321166
Stadtsparkasse Köln, Kto. 6342133, BLZ 37050198

Besuchen Sie uns auf der Hobbytronic in Dortmund · Besuchen Sie uns auf der Hobbytronic in Dortmund



Bild 1. Professional Page ist das Layout-Programm, welches Texte und Bilder zu fertigen Layout-Seiten verknüpft.

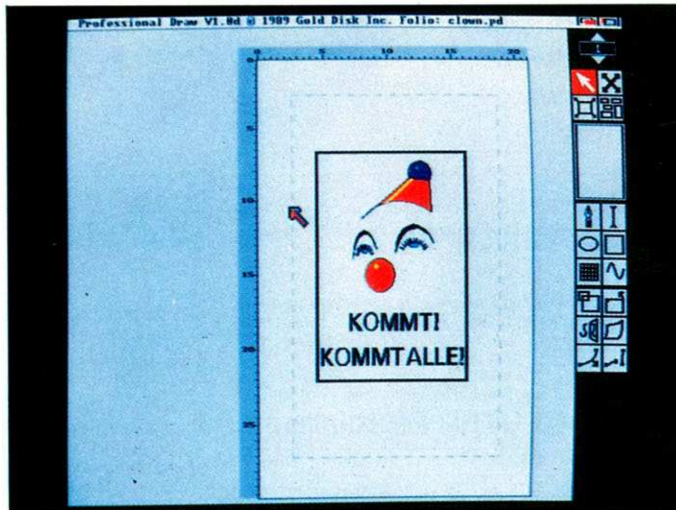


Bild 2. Professional Draw heißt das Zeichenprogramm, das voll auf den DTP-Betrieb zugeschnitten ist.

Durch Umwandeln der ursprünglichen Textfiles in das PostScript-Format.

Ein kleines PostScript-Beispiel-Programm zeigt Abbildung 1. Das Ergebnis, also das, was der Drucker unter Zuhilfenahme des PostScript-Interpreters daraus macht, können Sie in Abbildung 2 sehen. Der Aufbau der kleinen Datei entspricht fast der von bekannten Programmiersprachen, als Vergleich könnte am besten die Sprache 'FORTH' dienen. Das Besondere an der ganzen Sache ist, daß PostScript-Dateien mit einem beliebigen Editor eingegeben werden können; als einfachstes Beispiel kann hier der ED aus dem C-Verzeichnis stehen.

```
% PostScript-Testausdruck
% fur Apple-Laserwriter-Plus-kompatible
% PostScript-Emulationen

/Rechteck        % Prozedur definieren
{
        1 index 0 rlineto
                 % zweiten Parameter vom Stack holen
        0 exch rlineto
                 % ersten Parameter interpretieren
        neg 0 rlineto
                 % Rechteck fertigzeichnen
        closepath
} def

% Hauptprogramm

/ZapfChancery-MediumItalic findfont
                 % Schrift wahlen
36 scalefont     % Schriftgro3e 36p.
setfont          % setze aktuellen Font
newpath          % neuen Grafikweg offnen
0.0 setgray      % aktuelle Farbe: Schwarz
170 460 moveto   % "Stift" positionieren
175 120 Rechteck
                 % Aufruf der Prozedur
fill             % Flache fullen
stroke           % Grafikweg ausfuhren
1.0 setgray      % aktuelle Farbe: reines Wei3
215 535 moveto   % "Stift" positionieren
(Lies) show      % erste Textzeile schreiben
185 505 moveto   % "Stift" positionieren
(AMIGA) show     % zweite Textzeile schreiben
/Helvetica-Bold findfont
                 % neue Schrift wahlen
28 scalefont     % Schriftgro3e 28p.
setfont          % setze aktuellen Font
0.7 setgray      % 30 Prozent Grauraster
270 480 moveto   % "Stift" positionieren
(DOS) show       % dritte Textzeile schreiben
showpage         % Ausgabe der Seite auf Drucker
```

Abb. 1. Ein kleines PostScript-Programm, erstellt mit einem ASCII-Editor…

Nur eines ist wichtig, die PostScript-Dateien müssen reinen ASCII-Code enthalten,nur dann werden sie vom Interpreter auch ohne Fehler übersetzt. Natürlich ist es nicht Sinn einer solchen Anlage, vom Anwender zu verlangen, daß er sich erst mal mit PostScript auseinandersetzt, auch wenn dadurch der Nutzen des DTP noch steigt. Deshalb liegt dem Paket auch gleich eine passende Software bei: Professional Page und Professional Draw.

Vor dem Ausdruck steht die Erstellung – Der Anwender ist Redakteur, Layouter und Druckerei gleichzeitig

Professional Page ist das Layout-Programm, welches Texte und Bilder in die Form bringt, die schließlich per Laserdrucker ausgegeben werden soll. 'PPage' (so der abgekürzte Name) kann Texte verschiedener Art übernehmen. Darunter solche der bekanntesten Textverarbeitungsprogramme. Sinnvoller ist aber auf jeden Fall das Einlesen von reinen ASCII-Texten, die auf jedwede Steuerzeichen verzichten. PPage bietet genug Möglichkeiten, um den Text trotzdem individuell gestalten zu können.

Angefangen wird natürlich mit dem Aussehen der Seite. Bei unserem Test beließen wir es bei einer Standard-DIN A4-Seite, die als Testobjekt auf jeden Fall ausreicht. Diese Seite wird mit passendem Lineal (in verschiedenen Maßen darstellbar) auf den Bildschirm gebracht. Man kann diese Seite in allen Variationen betrachten, also Teilausschnitte der Seite oder Vergrößerungen oder Verkleinerungen.

Die angelegte Seite wird nun in Boxen unterteilt, wobei jede Box unterschiedlich gefüllt werden kann – entweder mit Text oder mit Grafik. Da PPage im hochauflösenden Modus arbeitet, lassen sich auch dementsprechende Bilder einlesen, seien sie digitalisiert oder mit Malprogrammen erstellt. In unserem Fall diente ein Demo-Bild des Malprogrammes 'Photon Paint' von MicroIllusions als Beispiel. Das Bild wurde komplett eingelesen und an die Größe der Box angepaßt. Der Text entstammt einem AMIGA-DOS-Beitrag und wurde als ASCII-Text eingelesen.

In puncto Bilder möchte man irgendwann selbst seine kreativen Fähigkeiten einsetzen und seinem Zeichentalent freien Lauf lassen. Dann bietet sich Programm zwei an: Professional Draw. Professional Draw ist ein Zeichenprogramm, das für die Arbeit mit

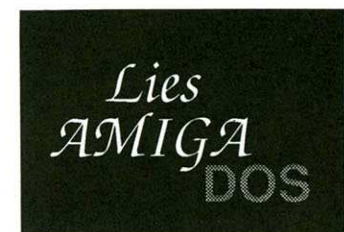


Abb. 2. …und interpretiert vom PostScript-Interpreter.


DMV
Daten- und Medienverlag

DTP konzipiert worden ist, PDraw ist nämlich objektorientiert. Während man bei Deluxe Paint quasi auf einer Leinwand zeichnet und diese auch nur komplett wieder säubern kann, ist es bei PDraw möglich, alle Elemente eines Bildes als Objekte anzusehen, die man nach Belieben verschieben, ersetzen, löschen oder übereinanderlegen kann. Jedes Objekt behält seinen Status, auch wenn das komplette Bild aus lauter Objekten zusammengefügt wurde. Und damit ist auch klar, warum PDraw der DTP-Anlage beiliegt: Mit PDraw lassen sich schnell Gestaltungselemente für DTP-Seiten erstellen.

Bei beiden Programmen ist allerdings eine gewisse Einarbeitungszeit vonnöten, so komplex sind sämtliche Funktionen.

Sind Grafik und Text bereit zur Übernahme, werden sie in die entsprechenden Boxen gesetzt. Hier nun kann die Feinarbeit beginnen. Einzelne Zeilen können zu Überschriften umgewandelt werden, Schriftarten lassen sich im Text anwählen, ganze Textpassagen können ein gänzlich anderes Aussehen bekommen.

Alles in allem ist jede Komponente des DTP-Pakets ein Erlebnis für sich; alles zusammen bildet eine Einheit, die es jedem ermöglicht, mit DTP praktische Erfahrungen zu sammeln und DTP konkret anzuwenden.

"Wer soll das bezahlen?" – Gedanken über eine mögliche Käufergruppe

Daß die DTP-Anlage nichts für den kleinen Geldbeutel ist, dürfte jedermann verständlich sein. Amiga 2500 mit 68020-Karte und eingebauter Festplatte sowie ein NEC 'Silentwriter LC890' (der allein schon über 12000,– DM kostet) plus Professional Page und Professional Draw für knapp 30000,– DM, da wird wohl kaum der 'Kleinanwender' in Frage kommen. Dies wäre so, als würde Handwerksmeister Schmidt für seine Fahrt zur Arbeitsstätte einen Jumbo-Jet benutzen.

Die eigentlichen Käufer sollen, wie schon gesagt, Menschen sein, die mit dem Erzeugnis 'DTP-Anlage' produktiv arbeiten. Produktiv heißt zum Beispiel Einsatz in der Werbebranche oder zu Präsentationszwecken. Und hier sind ein paar Punkte negativ anzukreiden:

Professional Page und Professional Draw mögen zwar auf dem Amiga sehr gut sein, der Amiga selbst macht aber in Punkto DTP dem PC noch keine große Konkurrenz. Gerade beim PC gibt es inzwischen eine erkleckliche Anzahl von sehr guten DTP-Programmen, mit denen Professional Page nicht so richtig mitkommt. Die Geschwindigkeit des Programms ist nicht sehr hoch, das dauernde Wechseln zwischen Menüleiste und Mauszeiger kann auf die Dauer ermüden. Auch beim Punkt Bildschirmdarstellung ist einiges zu bemängeln. PPage arbeitet im Interlace-Modus,

Der Desktop Publisher "Professional Page" im Test

Malprogramm als Sound-Editor

Wer hätte gedacht, daß man mit einem Malprogramm Sounds kreieren oder bearbeiten kann? Dieses Utility macht's möglich! SOUND - welch ein phantastisches Schlagwort! Die einen denken dabei sofort an den Hörgenuß. Nur einige wenige denken dabei auch an Wellen, Frequenzen, Hüllkurven und Amplituden. Doch gerade für diese Anwender ist dieses Programm von Nutzen, denn es bietet die Möglichkeit, kreativ eine Hüllkurve nach individuellem Geschmack zu konstruieren oder zu editieren.

Alles, was man benötigt, ist Dpaint3 oder ein ehrliches Malprogramm, welches nicht im Hold-And-Modify-Modus arbeitet, und schon kann's losgehen. Zuerst müssen zwei CLI-Prozesse aktiviert sein. Dies geschieht über den NEWCLI-Befehl, wenn bereits ein CLI-Prozeß über die Workbench oder ein SHELL aktiv ist. Von dem einen CLI-Fenster ruft man Wave.obj auf, von dem anderen danach DPaint3, indem man den Screen nach unten zieht oder mit <Amiga links + ESC> in den Hintergrund legt. Ist dies geschehen, so wählt man im



Setup ein Screen-Format von 640*256 Punkten mit 2 Farben. Die Arbeit mit dem Mahlprogramm verläuft wie gewohnt.

Als erstes sollte man im "Prefs"-Menü "Coords" wählen um immer über die graue Mausibraut Bescheid zu wissen. Wenn man berücksichtigt, daß die x-Achse für die Hüllkurve durch die Bildschirmpunkte (0,127) und (640,127), also waagrecht in der Bildschirm-Mitte verläuft, wird man feststellen, daß die Amplitude zwischen -128 (=0) und +128 (=255) liegen kann. Mit Sicherheit ist das kein Zufall, denn ein Sample besteht aus einer Folge von vorzeichenbehafteten 8-Bit-Werten. Da der Screen nun 640 Pixel breit ist, lassen sich auf ihm genau 640 Byte eines Samples darstellen. Nehmen wir einmal an, eine entsprechende Hüllkurve sei gezeichnet. Jetzt müssen wir mit F-10 die Menüleisten abschalten, damit sie

Abb. 3. Eine Testseite, erstellt mit Professional Page und mit Hilfe eines Demo-Bildes von PhotonPaint



Bild 3. Der NEC Silentwriter LC890, ein Laserdrucker der Sonderklasse. Grafik und Text in verschiedenen Schriftarten sind kein Problem, PostScript-fähig ist er sowieso

der zwar dank dem neuen Grafik-Chip und dem Multisync-Monitor nicht mehr das flackernde Bild liefert wie sonst, trotzdem werden die Augen beim Arbeiten ganz schön beansprucht. Auch das Positionieren innerhalb der Boxen ist recht mühsam, da die Amiga-Maus nicht gerade feinfühlig zu nennen ist. Das Zusammenspiel PDraw und PPage klappt reibungslos, der Amiga brachte es allerdings zustande, nach Fertigstellung einer Seite vor Ihrem Ausdruck mit einem 'Guru' auszusteigen. Zufall? Man kann es nicht genau sagen, jedoch ein unschöner Umstand für diejenigen, die sich die DTP-Anlage um des DTP willen gekauft haben und nicht, um die Exceptions des Betriebssystems kennenzulernen.

Den Amiga als Desktop-Publishing-Gerät zu sehen, ist eine neue Erfahrung. Ob er allerdings seinen stärksten Konkurrenten, den PCs mit ihrer jetzt schon starken Software, Paroli bieten kann, ist fraglich. Dazu müßten Soft- und Hardware noch genauer aufeinander abgestimmt werden – und eine größere Stabilität erreichen.

Trotz allem: Die Ergebnisse, die schon nach kurzer Einarbeitung zu erzielen sind, sind sehenswert...

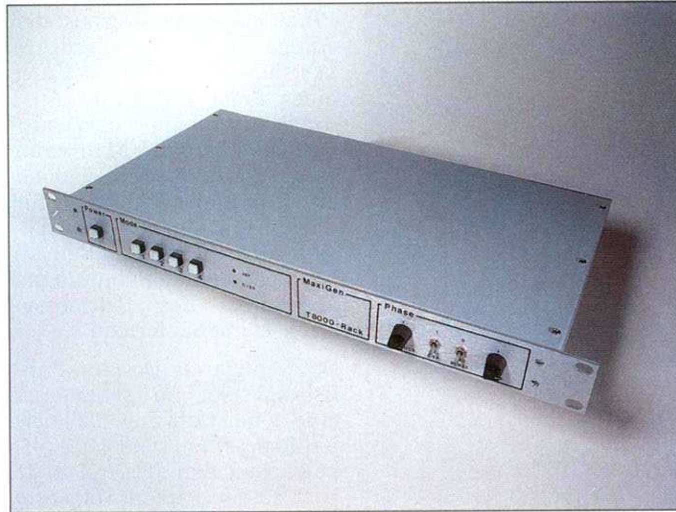
(Peter Schmitz/jb)

AMIGA DOS Info

AMIGA 2500 DTP

Das Amiga Desktop Publishing System basiert auf einem A 2500 mit 68020-Karte und 3 MByte RAM-Speicher. Fest eingebaut ist eine 44 MByte AutoBoot-Festplatte unter Kickstart und Workbench 1.3. Ausgeliefert wird die DTP-Anlage in mehreren Variationen, darunter mit 68030-Prozessor und Thermotransfer-Drucker; die uns zur Verfügung stehende wurde durch einen NEC-Laserdrucker 'Silentwriter LC890' mit eingebautem PostScript-Interpreter ergänzt. Beiliegende Programme sind: Professional Draw und Professional Page. Preis der Anlage: zirka DM 30000,–

Einsatzgebiet: Professionelle Layout-Erstellung.



Profi-Genlock

Grafik und Videoverarbeitung sind eines der interessantesten und vielseitigsten Themen auf dem Amiga. Und waren Genlocks, die in professioneller Qualität arbeiten, bis vor einem Jahr für den Heimanwender noch unerschwinglich, so hat sich diese Situation sehr zum Vorteil des Heimanwenders geändert. Begleiten Sie uns doch einfach auf unserem Streifzug durch die Welt der Bilder.

Wenn man sich eine Übersicht über die zur Zeit erhältlichen Genlocks verschafft, stellt man unweigerlich fest, daß doch große Unterschiede hinsichtlich der Bildqualität bestehen. Da vermag so manches Produkt nicht zu halten, was die aufwendig gestylte Verpackung verspricht. Ganz anders jedoch das Maxi-Gen von Merkens EDV, das mit Sicherheit zu den besten Systemen seiner Klasse gehört, soviel schon einmal vorweg.

Der Lieferumfang besteht aus dem eigentlichen Genlock Interface, das in einem soliden 19-Zoll-Einschubrahmen montiert ist, sowie einem Verbindungskabel, mit dessen Hilfe das Genlock zwischen den Amiga und den Monitor geschaltet wird.

Ein externes Netzteil versorgt das Genlock mit Spannung, so daß das Amiga-Netzteil nicht unnötig belastet wird. Ein Handbuch klärt die wichtigsten Fragen zur Inbetriebnahme und erläutert die Anschlußleiste, die an der Rückseite des Gerätes montiert ist. Der Anschluß an den Amiga, in unserem Fall fand ein A 2500 Verwendung, ist mit Hilfe des beiliegenden Kabels Momentsache. Platz findet das Genlock wahlweise in einem 19-Zoll-Rack, oder einfach auf der Zentraleinheit.

Auffallend sind einige Anschlüsse auf der Rückseite, die aufzeigen, daß das Maxi-Gen für den professionellen Bereich konzipiert wurde. Speziell die Anschlüsse <Ext Control>, <Ext Coder>, <Ref IP> und <Blk IP> zeigen auf, daß hier ein ausbaufähiges Grundgerät vorliegt, das mit entsprechender Peripherie zu einem kleinen, aber leistungsfähigen Fernsehstudio ausgebaut werden kann.

Eine genauere Ausführung der Möglichkeiten, die die eben genannten Anschlüsse bieten, würde sicherlich den Rahmen dieses Berichts sprengen, wir werden an anderer Stelle in einem Grundlagenartikel auf diese Thematik eingehen.

Um sinnvoll arbeiten zu können, ist darüber hinaus eine externe Videoquelle erforderlich, die mittels eines BNC-Anschlusses in das Genlock eingespeist wird. Hier kann eine Videokamera oder auch ein Videorekorder benutzt werden. Wollen Sie die Ergebnisse Ihrer Arbeit verewigen, so benötigen Sie einen zweiten Videorekorder, um Eingangssignal und Computergrafik zusammen auf einem Medium aufzuzeichnen. Soweit zu den Vorbereitungen, doch was läßt sich mit dem Genlock praktisch durchführen?

Jetzt kommt Bewegung ins Spiel

Neben praktischen Anwendungen wie das Versehen selbstgedrehter Videofilme mit Titel und sonstigen Credits, kommt das Genlock in dem Moment zur Geltung, wo eine Mischung mit Elementen eines Grafikprogramms erfolgt. Eine mit dPaint erzeugte Animation läßt sich beispielsweise problemlos in das laufende Videobild setzen. Damit sind Effekt- und Verfremdungsmöglichkeiten gegeben, die die Fähigkeiten

Technische Daten

Netzteil:

Primär:	220/240 Volt, 50Hz;
Sekundär:	15 Volt, 12 VA

Eingänge:

CVBS:	1V/ss
REF:	0,3 V Sync
0,3 V:	Burst
BLK:	2V/ss

Ausgänge:

RGB:	0,7 V/ss
Sync:	0,3 V/ss
CVBS:	1,0 V/ss

Ext Coder:

RGB:	0,7 V/ss
Sync:	0,3 V/ss
Key:	0,3 V/ss Sync + 0,7 V/ss Key

Alle Anschlüsse weisen eine Impedanz von 75 Ω auf.

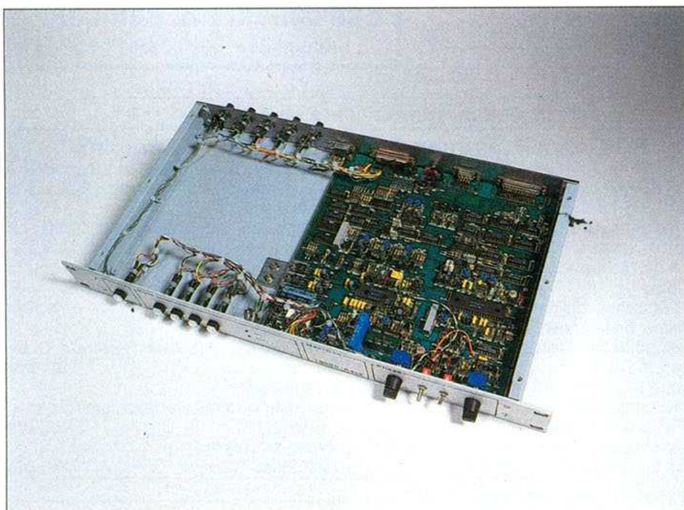


Bild 1. Das Innenleben des Maxi-Gen von Merkens EDV zeichnet sich durch übersichtliche, solide Verarbeitung aus

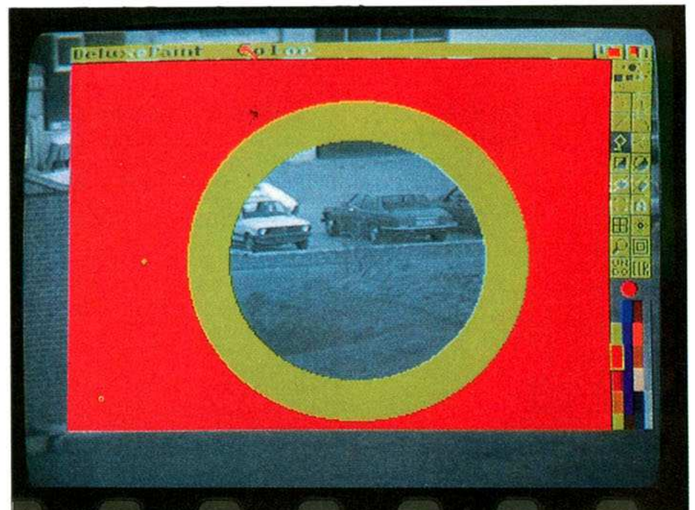


Bild 2. Die Kombinationsmöglichkeiten des Genlocks sind,...

des Amiga voll zur Geltung kommen lassen. Dabei ist es gleichgültig, in welcher Auflösung amigaseitig gearbeitet wird. HiRes wird ebenso problemlos integriert, wie Interlace oder eventuelle Overscan-Modes.

Praktisch sind dabei die "Mixschalter", mit denen festgelegt werden kann, in welcher Weise die Bildquellen verknüpft werden. Da wären die Zustände <nur Video>, <nur Amigagrafik>, <Amigagrafik über Videobild> und <Stanzen auf Farbe 0>.

Ergo kann praktisch zwischen dem Videoscreen und dem Grafikscreen umgeschaltet werden, was sowohl die Arbeit an der Grafik als auch die Kombinationen ungemein erleichtert. Stehen Videosequenz, Grafik und Animation, kann problemlos das Ergebnis aufgezeichnet und beispielsweise zu Präsentationszwecken gut verwendet werden. Dabei darf ein Umstand nicht übersehen werden. Das Signal, das die Videoquelle zur Verfügung stellt, kann nicht verändert werden, es ist also in dieser Konfiguration unmöglich, auf das Videobild amigaseitig zuzugreifen, da dieses Signal im Amiga ja nicht zur Bearbeitung zur Verfügung steht. Entsprechend muß also die Motivwahl für die Videoquelle erfolgen. Außerdem sollte darauf geachtet werden, die Motivgröße so zu wählen, daß eine sinnvolle Verknüpfung mit Grafikelementen noch möglich ist. Ein Firmenlogo auf der Stirn des Firmensprechers ist zwar sicherlich lustig, entspricht wahrscheinlich aber nicht den Vorstellungen über das fertige Produkt. Apropos Produkt: Die letztendlich als Mischung aufgezeichnete Kombination hat Fernsehqualität und braucht sich hinter den Ergebnissen manch professioneller Studios nicht zu verstecken, wobei das Endprodukt zum größten Teil von der Qualität der eingespeisten Signale abhängt.

Ob Präsentation, Bearbeitung von Urlaubsfilmen oder die Erzeugung eines Videoclips, die Möglichkeiten des Genlocks sind so vielseitig wie die Phantasie des Benutzers.

(mm)

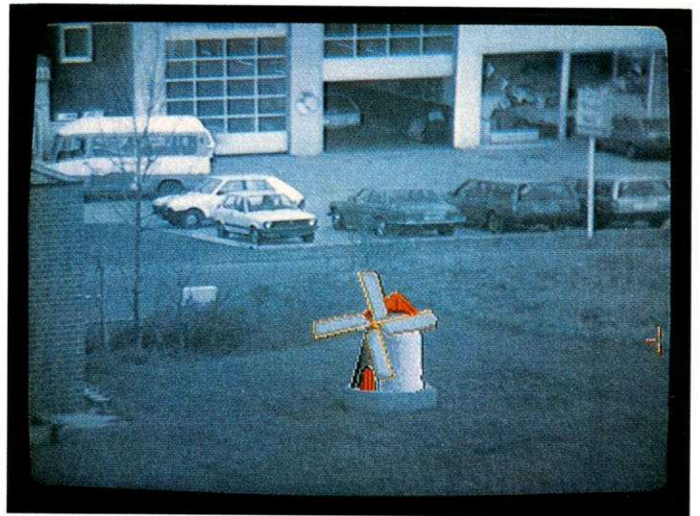


Bild 3. ...wie man sieht, mehr als vielseitig, und für Präsentationen bestens geeignet

Glossar

Doch was genau ist eigentlich ein Genlock-Interface? Wo liegen Unterschiede zu den ebenfalls populären RGB-Splittern bzw. Digitizern?

Keine Angst, hier sollen jetzt nicht theoretische Grundlagen besprochen werden. Ein kleiner Exkurs ist jedoch notwendig, um eventuelle Mißverständnisse im Keim zu ersticken.

RGB-Splitter

Da wäre zunächst der RGB-Splitter. Dieses Gerät hat die Aufgabe, ein Videosignal in seine einzelnen Farbkomponenten aufzusplitten: den ROT-, GRÜN- und BLAU-Anteil. Ergänzt werden diese drei Auszüge durch ein Synchronisations-Signal, um eine phasenrichtige Zuordnung der einzelnen Auszüge zu gewährleisten. Vorteil: Verarbeitet man die einzelnen Komponenten anschließend weiter, so sind eine höhere Farbechtheit und Detailtreue als bei der direkten Verarbeitung des Videosignals gewährleistet.

Ursache: Beim Videosignal werden alle Informationen einem HF-Träger aufmoduliert. Zur Rückgewinnung der Information muß das Signalgemisch demoduliert und beispielsweise in einer Matrix decodiert werden.

Das Signal, das letztendlich am Ausgang des RGB-Splitters abgegriffen werden kann, ist allerdings vom Amiga noch nicht direkt zu verarbeiten.

Verwendung findet der RGB-Splitter z.B. bei Standbild-Digitizern, um auf die Verwendung einer Farbfilterscheibe verzichten zu können.

Die Digitizer

Ein Digitizer dagegen hat die Aufgabe, ein ankommendes Video- bzw. HF-Signal in für den Computer verwertbare Spannungspegel umzusetzen. Dies wird in der Regel durch einen AD-Wandler erreicht und an der parallelen Schnittstelle in den Amiga eingespeist. Mit Hilfe entsprechender Software können nun die einzelnen Auszüge im Amiga wieder zu einem Gesamtbild zusammengesetzt und weiterverarbeitet werden.

Grundsätzlich muß bei Digitizern zwischen Realtime-Applikationen und Standbild-Digitizern unterschieden werden.

Genlock-Interfaces

Ein Genlock-Interface dagegen dient der Kombination zweier Bildquellen, in unserem Fall also der Überlagerung eines Videobildes und einer Computergrafik. Hört sich zunächst einfach an, gestaltet sich von der technischen Seite jedoch schon etwas anspruchsvoller, da sich die beiden zu vermischenden Signale weder in Phase, Amplitude noch Frequenz entsprechen. Auch mit der Synchronität ist es so eine Sache. Wenn beide Signalquellen nicht einen einheitlichen Takt aufweisen, muß zwangsläufig ein Signal "durchlaufen". Ein Effekt, der aus verstimmten Fernsehern ja hinlänglich bekannt ist. Zudem muß differenziert werden, an welchen Stellen sich die Signale überlagern und an welchen Stellen nicht. Schließlich sollen beide Bildquellen klar erkennbar sein. Transparenz heißt hier das Zauberwort, was nichts weiter bedeutet, als daß eine Farbe des Computerscreens als transparent oder Hintergrundfarbe erklärt wird. In der gemeinsamen Darstellung wird dann die Transparentfarbe zugunsten des eingespeisten Videobildes unterdrückt. Der Effekt: Alle Stellen der Computergrafik, die in der Hintergrundfarbe dargestellt werden, entfallen und werden durch das Videobild ersetzt. Als Beispiel mag hier der Vor- bzw. Abspann eines Filmes dienen, in dem einerseits das letzte Bild des Films, andererseits aber auch die Namen der Darsteller eingeblendet sind.

Soweit zur Begriffsentwirrung im Hardware-Dschungel.

AMIGA DOS Blitzlicht

Name: Maxi-Gen

Hersteller: Merkens EDV Computer-Video-Systeme Fuchstanzstr. 6A, 6231 Schwalbach, Tel.: (06196)-3026

Preis: 2789,– DM

Besonderheiten: Kann in Zusammenarbeit mit einem Mischer mit Superimpose-Fähigkeiten extern gesteuert werden.

Positiv:
- reichhaltige Anschlußmöglichkeiten
- professionelle Bildqualität
- solide 19-Zoll-Einschubtechnik
- externe Spannungsversorgung
- unkomplizierte Bedienbarkeit
- arbeitet mit allen Kameras gut zusammen
- einfacher, problemloser Aufbau
- sinnvolles, funktionales Bedienfeld
- gutes Preis/Leistungsverhältnis

Negativ:
- kein Hauptschalter
- Kanten des Einbaurahmens nicht entgratet

TFMX – Ein Soundgigant?

Ein Soundtool nicht nur für Profis

Vielen Amiga- und C64-Besitzern dürfte der Name Chris Hülsbeck schon ein Begriff sein, hat er doch in diversen Computerspielen für die entsprechende musikalische Untermalung gesorgt. Diesmal zeichnet er für einen Soundeditor verantwortlich, der sich wirklich sehen lassen kann.

Der Amiga ist ein Computer mit leistungsstarken Soundfähigkeiten. Er ist in der Lage, digitalisierte Töne abzuspielen, ohne den Hauptprozessor zu belasten. Weiterhin kann er vier Klänge gleichzeitig auf zwei Kanälen wiedergeben. Doch ist es beim Amiga wie bei "normalen" Musikinstrumenten: Bevor man in den Genuß kommt, die hervorragenden Musiken zu Gehör zu bekommen, muß man in der Lage sein, das Instrument zu bedienen. Bei dem Amiga ist es da etwas einfacher, wenn man sich eines Sound-Editors bedient. Wir haben uns wieder einmal auf dem Softwaremarkt umgesehen und einen Sound-Editor unter die Lupe genommen, der die Erstellung von Musikstücken erleichtert.

Das TFMX-Soundtool ist ein Hilfswerkzeug zur Musikprogrammierung, das auf den ersten Blick durch seine Vielzahl an Optionen etwas unüberschaubar wirkt. Hat man sich jedoch erst einmal mit der Bedienung dieses Editors vertraut gemacht, entdeckt man Funktionen, von denen sich so manches Soundtool eine Scheibe abschneiden könnte. Allein schon die Tatsache, daß der "Komponist" über drei unterschiedliche Editoren (Pattern-, Track- und Makro-Editor) verfügt, macht diese Software zum individuellen Musik-Tool.

TFMX verarbeitet nur Samples, also digitalisierte Geräusche – die Amiga-internen Sounds werden ignoriert. Dies ist jedoch vom Klangtechnischen kein großer Nachteil, da die internen Töne des Amiga ohnehin für Musiker von untergeordneter Bedeutung sind.



Das TFMX-Soundtool wartet nicht nur mit hervorragenden Soundeffekten, sondern auch mit beispielhafter Bedienungsfreundlichkeit auf

Lediglich der Speicherplatz wird durch die umfangreichen Samples beeinträchtigt.

Wobei wir auch schon bei der Speicherbelegung des TFMX wären. Der Benutzer wird nach dem Booten der Diskette aufgefordert, die entsprechende Rechnerkonfiguration (512 kByte bzw. 1 MByte) anzugeben. Speichererweiterungen auf über 1 MByte werden zur Zeit noch nicht unterstützt.

Für Rechner mit einer Speicherkapazität von 512 kByte stehen 25 kByte Musikdaten zur Verfügung, während diejenigen, die einen Amiga 2000 bzw. eine Speichererweiterung besitzen, auf 50 kByte Musikdaten zugreifen können.

Dabei unterscheidet TFMX innerhalb dieser beiden Konfigurationen noch zwei weitere Darstellungsformen (16 und 4 Farben). Wählt man die Vierfarb-Darstellung, stehen den 512-kByte-Rechnern 225 kByte und den 1-MByte-Amigas 355 kByte Samplespeicher an, im 16-Farben-Modus sind es dagegen nur 145 bzw. 275 kByte.

Die Erstellung eines Musikstückes fängt mit dem Einladen der Samples an. Nebenbei bemerkt: Das TFMX-Soundtool wird mit drei Disketten ausgeliefert, wobei auf zwei Disketten nur Sound-Samples im IFF-Format zu finden sind. TFMX akzeptiert verschiedene Sampleformate wie IFF, DATA und Future Sound.



Der Vierfarb-Editor – eine etwas unübersichtliche Darstellung, dafür jedoch mehr Platz für Samples

Patterns im Viervierteltakt

Nachdem die "Instrumente" in Form von Samples (bis zu 256 Samples möglich) bereitgestellt sind, gilt es, den Takt einzustellen. Hier hilft eine kleine Besonderheit des Editors – das Metronom, also der Taktgeber, das wahlweise mit einer gerade abspielenden Melodie parallel laufen oder auch ausgeschaltet werden kann. Dieses Metronom ist frei einstellbar und verwendet nicht die Soundkanäle.

Nun stehen dem Benutzer zwei Möglichkeiten der Soundererzeugung zur Verfügung: zum einen das direkte Einspielen von Musikstücken per Klaviatur und zum anderen die Programmierung. Gehen wir zunächst den einfacheren (und schnelleren) Weg, nämlich den der Musikerzeugung per Klaviatur.

TFMX verfügt über eine Record-Page, womit man über die Tasten des Amiga Musikstücke spielen kann. Die gespielten Stücke können abgesichert und nachträglich bearbeitet werden. Hier stellt das Metronom ein nützliches Werkzeug zur Verfügung. Es lassen sich sogar Auftakte als Vorzähler vorgeben. Doch gehen wir lieber auf den interessanteren Teil des TFMX ein, dem Programmieren von Kompositionen.

Die Samples stehen ja schon zur Verfügung, fehlen eigentlich nur noch die Noten und die Toneffekte. Dazu sehen wir uns die drei Editoren etwas genauer an. Im Pattern-

AMIGA DOS Blitzlicht

Name: TFMX-Editor
Hersteller: Demonware
Quelle: Fachhandel
Preis: zirka 130 DM

Positiv:
- unterschiedliche Sample-Formate einlesbar
- Musikstücke können in jede Programmiersprache eingebunden werden
- Festplattenunterstützung
- gutes Handbuch

Negativ:
- zur Zeit noch keine MIDI-Unterstützung



In der Record-Page kann man Musikstücke direkt über die Tastatur eingeben. Dabei sind auch Auftakte kein Problem

Editor werden einzelne Teile einer Melodie vorgegeben. Diese Mustervorgabe kann beispielsweise ständig wiederholt werden, wie man dies von Baß- und Schlagzeugbegleitung her kennt. Man muß also nicht immer wieder den gleichen Passus eingeben, wenn bestimmte Routinen verwendet werden. In den Patterns stehen die Noten und Statussymbole eines Musikstückes, die Makros – dazu gleich mehr – anwählen. Ein Pattern entspricht dabei einem Takt.

Wie bei den anderen zwei Editoren erleichtert die Cut-Copy-Paste-Funktion das Editieren ungemein. Diese Funktion sollte jedem Amiga-Besitzer ja vom Amiga-BASIC her bekannt sein.

Was ist eigentlich ein Makro? Nun, ein Makro ist ein Programm, das ein Sample aufruft. In dem Makro-Editor werden also die Samples für ein Musikstück verwaltet. Der TFMX läßt bis zu 128 Makros, die in der Makro-Page editiert werden, zu. Grob gesagt ist die Makro-Page eigentlich nichts anderes als ein Editor für eine spezielle Programmiersprache, die Makroprogrammierung eben.

Der dritte Editor ist der sogenannte Track-Editor, in dem die einzelnen Patterns aufgerufen werden. Diese Patterns können an dieser Stelle transponiert werden. Beim TFMX stehen acht Tracks zur Verfügung, wobei natürlich eine kleine Einschränkung gemacht werden muß, da der Amiga ja nur vier Audio-Kanäle besitzt, also nur vier Töne gleichzeitig abspielen kann. Die weiteren vier Tracks erlauben jedoch eine flexiblere und dadurch auch komplexere Gestaltung von Musikstücken.

Da capo – aber al fine!

Was kann der TFMX nun eigentlich alles? Nun, auf die verschiedenen Sample-Formate sind wir ja schon eingegangen. Die gängigsten Effekte wie Portamento, Fading, Vibrato, Echo etc. sind natürlich enthalten. Ferner unterstützt er die Multifunktionsfähigkeit des Amiga. Musikstücke können auch aus einzelnen Teilstücken zusammengefügt werden, was die Programmierung etwas leichter macht. Gleichzeitig sind bis zu 32 Songs verwaltbar. Als weiteres Positivum fällt auf, daß das Programm, das übrigens nur mit einem Hardware-Zusatz (Dongle) läuft, zwei Festplatten unterstützen kann.

Will man bereits erstellte Songs in eigene Programme einbauen, so ist dies auch kein Problem. Auf der Diskette ist ein Player vorhanden, der ermöglicht, daß Musikdaten auch eigenständig lauffähig sind. Das deutsche Handbuch ist ausführlich und gibt eine ausreichende Einführung anhand kleinerer Beispiele in den Umgang mit dem Editor. Ferner lassen sich Datendisketten vom Editor aus erstellen.

Leider ist das Programm noch nicht MIDI-fähig, es wurde jedoch angekündigt, daß das Update diese Option unterstützen soll. Ferner soll laut Hersteller ein Software-Sampler implementiert werden.

(br)

Edgar Meyzies

Überzeugendes Mathepaket

MathTreasures-2.0

Seit Anfang des Jahres wird eine Mathe-Bibliothek in Modula-2 (M2AMIGA) angeboten, die eine wertvolle Ergänzung zu bereits veröffentlichten Modulen darstellt.

Was soll das? Nur eine Diskette, kein Handbuch und dafür über 100 DM berappen, das wirkt doch überzogen! Weit gefehlt! Die Diskette weist eine (gepackte) Dokumentation auf, die einen Drucker über 120 Seiten (einschließlich der Programmtexte der Schnittstellenmodule) ausspucken läßt. Was wird denn da eigentlich dokumentiert? Das Paket umfaßt 78 kurze Module, die die in Tabelle 1 dargestellten Bereiche abdecken. Die hohe Anzahl der Module besagt für sich alleine nicht viel, weder im Hinblick auf den Umfang der implementierten Funktionen, noch läßt sie Rückschlüsse auf die Handhabbarkeit zu. Bei näherer Betrachtung entdeckt man schnell, daß es sich eigentlich nur um wenige, thematisch zusammenhängende Modulpakete handelt, die jeweils auf die Arbeit mit einem Datentyp (zum Beispiel FFP, REAL, LONGREAL) ausgelegt sind. Die hohe Anzahl der Module ergibt sich also aus der Spezialisierung der Module auf Datentypen, aber auch aus dem Bemühen des Programmautors um einheitliche Schnittstellen für die Funktionsprozeduren. Sie verrichten die gleichen Aufgaben, nur mit jeweils einem anderen Datentyp. So wird bei der Programmentwicklung der Wechsel zwischen Datentypen erleichtert, um zum Beispiel die Rechengenauigkeit zu erhöhen oder bewußt auf “speed“ zu trimmen.

Die Implementierung der mathematischen Funktionen erfolgte sehr systematisch und konsequent. Allgemein hervorzuheben sind die anwenderfreundlichen Benutzerschnittstellen, die auf dem Amiga Bewährtes fortführen und, wo immer möglich, in der Mathematik übliche Notationen zulassen. Das Paket ist durchaus für “ernsthafte“ (frei nach dem Vornamen des Entwicklers E. Heinz) Anwendungen geeignet. Es schließt eine Lücke und macht dadurch Modula-2 auf dem Amiga noch attraktiver.

Zur Dokumentation könnte man auch die Texte der Implementationsmodule rechnen, die ausgedruckt nochmals etwa 200 Seiten füllen.

Ohne hohen Aufwand könnte die Beschreibung der Schnittstellen erheblich verbessert werden. Um professionellen Anforderungen zu genügen, sollten die Leistungsgrenzen der Module angegeben werden.

Die Quelltexte sind bestens geeignet, die effiziente Programmierung mathematischer Funktionen zu studieren. Die Module sind hervorragend strukturiert und sauber programmiert. Der Portierbarkeit scheint ein hoher Wert beigemessen zu sein. Die Übertragung der Matrix-Module auf einen anderen Rechner war innerhalb weniger Stunden abgeschlossen.

Herauszustellen ist auch die hohe Zuverlässigkeit der Module und ihre Robustheit gegenüber fehlerhaften Eingaben. Die Programmlaufzeiten haben positiv überrascht. Coprozessoren werden optimal genutzt.

Schade, daß die Mathe-Bibliothek (noch?) nicht als “Library“ im Sinne des Amiga implementiert ist und somit nicht von allen Sprachen aus nutzbar ist. Auch auf der Ebene der Objektdateien kann man sich nicht treffen, weil der Compiler des M2AMIGA (wie lange eigentlich noch?) sich die Exklusivität eines eigenen Linkformates leistet.

Bruchrechnung

Bruch haben Sie gewiß nicht einzukalkulieren, wenn Sie die “MathTreasures“ für diesbezügliche Rechnungen einsetzen. Sieben Modulpaare unterstützen diese häufig belächelte Form der Mathematik. (Haben Sie schon einmal an die Genauigkeit der Bruchrechnung gedacht?) Brüche werden als Zeichenketten eingelesen und so auch wieder ausgegeben. Alle üblichen Arten der Bruchrechnung (Addition, Substraktion, Multiplikation, Division, GGT, KGV, Kürzen, Kehrwert usw.) werden unterstützt. Dabei kann mit unterschiedlicher Genauigkeit gerechnet werden, nämlich mit 16 und mit 32 Bit breiten (INTEGER und LONGINT) Bruchzahlen. Es wurde echte Integer-Arithmetik implementiert. Brüche können sowohl mit Bruchstrich als auch als Dezimalbruch (Dezimalzahl, als Bruch interpretiert) eingegeben werden. Sie erraten sicherlich schon die Struktur des Datentyps “Bruch“; er ist wie eine Zahl des Typs “REAL“ mit einer Breite von 32 Bits implementiert, als varianter Verbund. An der Stelle des Dezimalpunktes kann auch der Bruchstrich vorhanden sein.

Nur keine komplexen...

... Zahlen, bitte! Wer aber auf technisch-wissenschaftlichem Gebiet arbeiten muß, der findet in dem Paket “Complex“ neun äußerst leistungsfähige Module für komplexe Zahlen der Form

“Realteil + Imaginärteil * i“.

Besondere Ein- und Ausgaberoutinen ermöglichen den Dialog mit dem Programm. Die Funktionen sind auf reelle Zahlen der Typen FFP, REAL und LONGREAL spezialisiert.

Neben den mathematischen Grundfunktionen (wie Addition, Multiplikation usw.) wurden weitere Funktionen (zum Beispiel Umwandlung in Polarkoordinaten und umgekehrt, Operationen komplexer Zahlen mit Gleitpunktzahlen, Kehrwertberechnungen usw.) implementiert. Es ist auch möglich, komplexe Zahlen aus Ausdrücken (reelle Zahlen) zu erzeugen. Alle mathematischen Ausdrücke wurden als echte Funktionsprozeduren implementiert, um eine Schachtelung in gewohnter Weise zu ermöglichen.

Bruchrechnung:
Ein- und Ausgaben, alle arithmetischen Operationen, Konvertierung von Brüchen in Gleitpunktzahlen und umgekehrt

Rechnen mit komplexen Zahlen:
Ein- und Ausgaben, alle üblichen Standardfunktionen für komplexe Zahlen der Form “Realteil + Imaginärteil * i“, implementiert für FFP, REAL und LONGREAL

Umwandlung von Fließkommazahlen:
Typenübergänge zwischen FFP, REAL und LONGREAL

Logische Funktionen:
Ergänzung der logischen Funktion NOT um die von BASIC her bekannten

Ergänzungen/Erweiterungen der Math-Libs:
Komplettierung der MathLib0 durch eine neue MathLib1, die im wesentlichen zusätzliche Winkel- und Logarithmik-Funktionen bietet (getrennte Module für die Datentypen FFP, REAL und LONGREAL) sowie die Leistungsfähigkeit der jeweiligen Hardware voll ausschöpft.

Arbeiten mit Matrizen:
Für die drei Datentypen (FFP, REAL, LONGREAL) jeweils 13 Module, um Matrizen beliebiger Größe anzulegen und damit zu rechnen. Dazu gehört auch das Lösen linearer Gleichungssysteme.

Tabelle 1. MathTreasures-2.0 bietet für wichtige Bereiche der Mathematik eine reichhaltige und sichere Implementation leistungsfähiger Funktionen

In der Modulvielfalt des Paketes geht "FloatConv" fast unter, obwohl es schon bei einfachen Anwendungen von geschwindigkeitsmäßig hohem Nutzen sein kann.

Verflüssigen

Unter Ausschöpfung von Möglichkeiten der Library "MathDoubTrans" wurden sechs Prozeduren zur Umwandlung von Fließpunktzahlen (FFP < – > REAL, FFP< – >LONGREAL und LONGREAL< – >REAL) zur Verfügung gestellt. Die Besitzer mathematischer Coprozessoren werden über die neuen Möglichkeiten besonders erfreut sein.

Unkraut im Gerstenfeld?

Auch dem Modul "LogicLib" droht das Schicksal, übersehen zu werden. Das Modul stellt fünf Prozeduren (EQV, IMP, NAND, NOR und XOR) zur Verfügung, die der Compiler "m2c" nicht kennt. Wie mag das Modul nur in die Mathe-Bibliothek geraten sein, etwa wie eine Kornblume in ein Gerstenfeld? Nützlich ist es allemal!

Genau 15 Funktionsbibliotheken sind es, die die unter M2AMIGA implementierten Mathe-Module ergänzen bzw. sogar ablösen wollen. Die hohe Anzahl mag erschrecken, hatte man es doch bisher mit wesentlich weniger Modulen zu tun.

X-mal Math

Bei näherer Betrachtung erkennt man schnell, daß es eigentlich nur vier Gruppen von Modulen sind, die jeweils sehr ähnliche Aufrufkonventionen aufweisen:

○ Ersatz der bereits bestehenden Standardmodule

- "MathLib0"
- "MathLibFFP" und
- "MathLibLong"

um die Sicherheit im Gebrauch zu erhöhen bzw. die Hardware (Mathe-Coprozessor) voll auszureizen. Die Anleitung enthält klare Empfehlungen zum Gebrauch. Die Library "Mathieeedoubtrans" wird sehr geschickt eingesetzt, um an Tempo zuzulegen.

○ Ergänzung der MathLib0 durch Implementierung zusätzlicher Standardfunktionen (zum Beispiel: deg, rad, rez, sqr, log dualis, Zehnerlog., Log. zu beliebiger Basis, pow, tan, cot, arcsin, arcos, arccot, sinh, cosh, tanh, coth, arcsinh, arccosh, artanh, arcoth) jeweils für die Datentypen FFP, REAL und LONGREAL.

○ Schneller Ersatz für die Standardbibliothek "MathLib0" durch Verzicht auf Überprüfung von Argumenten bei Aufruf der implementierten Funktionen und zur Ausschöpfung der Vorteile, die Mathe-Coprozessoren bieten.

Die "Mathxxx-Module" zeichnen sich durch ihre Fehlerbehandlung aus. Aussagen zur Rechengenauigkeit fehlen jedoch.

Im Viereck springen...

möchte man, wenn der "DIR-Befehl" 39 Namen von Modulen zur Arbeit mit Matrizen auf den Bildschirm knallt. (Ein Paukenschlag wäre angemessener, wie Sie gleich sehen werden.) Es sind jedoch eigentlich nur 313 Module, jeweils für die Datentypen FFP, REAL und LONGREAL implementiert. Nur wenige Programmiersprachen beherrschen die Matrizenrechnung. Für Modula-2 besteht nun eine äußerst effiziente Implementierung, die leicht auf andere Systeme zu portieren ist. Herausragend ist die dynamische Verwaltung der Matrizen, die beliebige Größen zuläßt und den Arbeitsspeicher geschickt nutzt. Das haben wir natürlich mit sehr großen Matrizen ausprobiert. No problem! Durch Verwendung dynamischer Datenstrukturen konnte eine besonders hohe Geschwindigkeit für die Ausführung der Matrixoperationen erzielt werden. Zuerst traut man seinen Augen nicht, wenn eine Matrix mit 100x100 Feldern ohne Mathe-Coprozessor für FFP-Zahlen in 1:52 Minuten invertiert wird. Für REAL-Zahlen ergaben sich 3:15 und für Zahlen des Typs LONGREAL 3:32 Minuten. Beeindruckend ist auch die mitgelieferte "3D-Demo", die mit Matrizenoperationen arbeitet.

Die Basismodule lassen eine Struktur erkennen, die auf Erweiterung ausgelegt ist. Alle Vorbereitungen sind getroffen, um beliebige Elementtypen zu implementieren. Wie wäre es zum Beispiel mit Vektoren oder Polynomen? Ein nächstes Update wird es erweisen.

Die Matrix-Module bieten die in Tabelle 2 dargestellte Funktionalität. Anschauliche Demoprogramme zeigen, wie unkompliziert es ist, die angebotenen Funktionen zu nutzen. Dazu eine kleine Kostprobe: Um eine Matrize anzulegen, sind nur zwei Prozeduren einzusetzen:

● Mit dem Aufruf von **"InitMatrix(testMatrix)"** werden für "testMatrix" (vom Datentyp MATRIX) die Basisstrukturen angelegt, um die übrigen Funktionen darauf anzuwenden.

● Der Aufruf **"DimAndFillMatrix(testMatrix,z,s,0.0)"** bewirkt, daß unser Testobjekt mit "z" Zeilen und "s" Spalten angelegt sowie jedes Element mit "0.0" initialisiert wird.

Auf die "LGSLib" (Lineare Gleichungs-Systeme) möchten wir Ihr besonderes Interesse richten. Das Modul stellt sehr mächtige Routinen zum Lösen praktisch beliebig großer linearer Gleichungssysteme zur Verfügung. Die Routinen bewährten sich in volkswirtschaftlichen Simulationsmodellen. Sie lieferten stets korrekte Ergebnisse und überzeugten durch ihre hohe Arbeitsgeschwindigkeit.

Schätzchen oder Schatz?

Nach mehrtägiger Arbeit mit vielen der angebotenen Funktionen fällt die Bewertung der "MathTreasures" mit "echter Schatz" sehr leicht. Faszinierend ist die Leistungsfähigkeit der "Complex-" und der "Matrix-Module". Bereits als Schüler und als Student hätte ich gerne ein solches Paket besessen. Heute unterstützt es mich im Bereich der Simulation auf ausgezeichnete Weise.

(mm)

Modul	Enthaltene Funktionen
"MatrixArith":	arithmetische Standardoperationen
"MatrixBase":	Matrixerzeugung und -verwaltung
"MatrixCopy":	Kopieren von Matrizen und Matrixteilen
"MatrixDecs":	zentrales Deklarationsmodul
"MatrixDiskIO":	Laden und Speichern von Matrizen auf Diskette
"MatrixElemOps":	Operationen auf allen Matrixelementen
"MatrixIO":	I/O für Matrizen auf Grundlage von "InOut"
"MatrixMinMax":	Minimum-/Maximumbestimmung
"MatrixNorm":	Normberechnung für Matrizen
"MatrixOps0":	diverse Zeilen- und Spaltenoperationen
"MatrixOps1":	Gauß-Algorithmus, Transponierung, Determinanten und Rangbestimmung
"MatrixOps2":	LR-Zerlegung und Matrixinvertierung
"LGSLib":	Lösen beliebiger linearer Gleichungssysteme

Tabelle 2. Für jeden der drei Datentypen FFP, REAL und LONGREAL bietet "MathTreasures-2.0" mächtige Funktionen zur Bearbeitung von Matrizen

AMIGA DOS Blitzlicht

Name: MathTreasures-2.0
Hersteller: A + L AG, Schweiz
Vertrieb: Fachhandel
Preis: 100,– DM

Positiv:
- Installation auf Festplatte möglich
- Einbinden in eigene Programme
- hohe Genauigkeit
- sehr schnell
- ausführliche Dokumentation
- einheitliche Schnittstellen
- gutes Preis-Leistungsverhältnis

Negativ:
- noch nicht als Library implementiert
- eigener Linker

Sendung läuft!

Von Computer zu Computer – Kermit hilft

Bevor Sie jetzt die Nase rümpfen, weil Sie glauben, wir wollten Ihnen etwas über die Sesamstraße erzählen, ist es besser, Sie über Kermit aufzuklären: Kermit ist ein Public-Domain-Programm, daß unter Zuhilfenahme eines seriellen Nullmodem-Kabels Daten über die serielle Schnittstelle überträgt. Wie, Sie verstehen nur Bahnhofshalle, Ausgang Nord? Na, dann lesen Sie doch mal weiter, das Ganze hat nämlich einen Sinn.



Unter den Amiga-Besitzern gibt es viele, die schon vorher einem Computer das Programmieren beigebracht haben, viele 8-Bit-Geräte stehen seit dem Erwerb des Amiga als Requisite oder Museumsware ungenutzt in der Ecke. Kein Problem, wenn da nicht die Datei über die aufgenommenen Videofilme oder die Dia-Sammlung auf einer zugehörigen Diskette existieren würde, die Freund(in) Amiga absolut nicht lesen will. Was tun? Den kleinen Vorgänger an die aufsteigende Computer-Generation verschenken (dreijährige Kleinkinder sind geradezu glücklich, wenn sie ein Ex-High-Tech-Gerät auseinandernehmen dürfen), die entsprechenden Disketten zu diversen Weihnachtsgeschenken umbauen (Diskettenuhren, Disketten-Spardosen, Disketten-Schlipse?!)? Eigentlich schade, wenn man bedenkt, wieviel Arbeit und Zeit damals investiert wurden, um sich mit dem Thema Computer vertraut zu machen.

Bei manchen Amiga-Besitzern kann es aber auch sein, daß sie beruflich und privat mit den 'Konkurrenten' des Amiga zu tun haben, nämlich ST und PC. Bei beiden gibt es einen großen Hinderungsgrund, Daten und Dateien von den jeweils anderen Disketten zu lesen – das Diskettenformat. Und jetzt kommt Kermit ins Spiel – natürlich nicht der Frosch, sondern das Public-Domain-Programm. Und die serielle Schnittstelle unseres Amiga, die bisher den Aschenputtel-Status besaß, bekommt plötzlich einen ganz neuen Stellenwert.

Noch ein Bit und noch ein Bit – seriell ist nicht schnell

Um Kermit sinnvoll nutzen zu können, brauchen wir noch ein klitzekleines bißchen Hardware – das Nullmodem-Kabel. Bingo!

Bei den meisten Lesern wird gerade noch das Wortfragment 'Modem' irgendwo im Hinterkopf irgendeine Assoziation hervorrufen, bei der Konstellation 'Nullmodem' wird jedoch so mancher erst einmal erstaunt aufschauen, das Wortende 'Kabel' erzeugt dagegen bei 'Nichtbastlern' Schauderwellen. Dabei ist das alles gar nicht so wild, wie es sich bis jetzt angehört hat.

Bei einer 'normalen' Datenübertragung wird jedes Byte, das vom Sender zum Empfänger geschickt wird, durch eine bestimmte Reihenfolge von Parameter-Bits umgeben. Damit läßt sich erreichen, daß der Empfänger dem Sender mitteilt: "O.K. alles richtig angekommen!", oder "Das war wohl nix! Nochmal!!". Ein solcher Vergleich der abgesendeten Daten mit den empfangenen heißt in der Fachsprache 'Übertragungsprotokoll'.

Heutzutage wird viel über DFÜ gesprochen (Datenfernübertragung), und diese DFÜ wird in der beschriebenen Art benutzt. Bekannteste Möglichkeiten der DFÜ sind Datenübertragung über Modems (MOdulator – DEModulator) oder Akustikkoppler. Im Grunde genommen findet hier ein Kreislauf statt, wenn Sender und Empfänger ihren Status wechseln, der Sender also auch als Empfänger und der Empfänger auch als Sender benutzt werden kann. Es geht auch anders, man kann Sender und Empfänger bestimmen, dies soll uns jedoch hier nicht interessieren.

Bei den Sendern findet also eine Umwandlung unserer Daten in eine Bitfolge statt, damit diese korrekt zum Empfänger gesendet werden können. Der wiederum wandelt die empfangenen Daten in normale Bytes um – wenn nicht bei der Übertragung etwas danebengegangen ist. Wie schon gesagt, wir können uns das Ganze als einen Datenkreislauf vorstellen, in dem zwei Computer über ihre serielle Schnittstelle und Umwandlungsgeräte miteinander kommunizieren.

Warum aber seriell und nicht parallel? Modems und Akustikkoppler nutzen eine Verbindung aus, die wir ebenfalls benutzen, das Fernmeldenetz. Dieses besteht im Grundprinzip nur aus zwei Drähten, der Datenleitung und der Masseleitung. Würden wir parallel übertragen, wären schon acht Datenleitungen erforderlich, um alle Zeichen nutzen zu können. Erinnern wir uns: Jedes Byte hat eine Länge von acht Bit, jeder Zeichencode läßt sich in einem Byte unterbringen. Damit haben wir die Möglichkeit, 255 Zeichen in einem Byte (2↑8 = 256-1 = 255) unterzubringen. Da jedes Bit einen Stellenwert hat (2↑0, 2↑1, 2↑2, usw.), brauchen wir für jeden Stellenwert auch eine Leitung. Das geht aber nicht, denn, wie gesagt, wir haben nur zwei. Splitten wir die Bytes in Bits auf und schicken sie hintereinander los, brauchen wir tatsächlich nur zwei Drähte (theoretisch) und ein Gerät, was aus den zerstückelten Bytes wieder ganze macht.

Da gibt es nur eine Schwierigkeit. Der 'Zusammensetzer' muß wissen, ob das, was er zusammensetzt, auch so richtig ist. Er muß sich also beim Sender erkundigen, ob das zusammengesetzte Byte genauso aussieht wie das zerstückelte. Und jetzt kommt das Übertragungsprotokoll zur Geltung. Es bringt quasi als Bote Informationen über die gesendeten und empfangenen Bits vom Empfänger zum Sender und sorgt damit für einen reibungslosen Ablauf.

Der Ablauf bei der seriellen Schnittstelle sieht also vereinfacht so aus:

Sender -> Daten -> Umwandler (Modem) -> Kabel -> Umwandler Empfänger -> Protokoll -> Sender -> entweder o.k. oder neuer Versuch -> Empfänger.
Und jetzt haben wir einen gravierenden Nachteil der seriellen Datenübertragung entdeckt: Sie ist zu langsam. Gleichzeitig finden wir aber auch den großen Vorteil: Fernmeldenetze gibt es überall auf der Welt. Einen Datenbestand von Hamburg nach New York zu schicken, kostet höchstens ein müdes Lächeln (und Telefongebühren, bitte weiter lächeln).



Die Datenfernübertragung mit wenigen Worten zu behandeln, ist ausgeschlossen, dazu ist das Thema zu umfangreich. Für unsere Zwecke jedoch soll es erst einmal reichen, denn wir brauchen nicht allzuviel Verständnis der Hardware beim Umgang mit Kermit.

Was überträgt Daten ohne Modem? Ein Nullmodem-Kabel

Mit den vorgestellten Geräten haben wir quasi einen kompletten Datenkreislauf errichtet (auch wenn nicht unsere Daten zurückkommen, sondern andere). Aus diesem Kreislauf können wir jetzt zwei Teile entnehmen: das Modem und das Übertragungsmedium, also das Fernmeldenetz. Übrig bleiben nur die Anschlußkabel am Computer. Diese zwei könnten wir jetzt miteinander verbinden und schon...
...gibt's Kurzschluß – oder besser gesagt, Schwierigkeiten. Mit der Zwei-Draht-Theorie ist das nämlich so eine Sache. Nach dem Modem wäre es ja kein großes Problem (obwohl es auch da mehr als zwei Drähte gibt, aber das ist Inhalt eines anderen Artikels). Vor dem Modem jedoch müssen wir unterscheiden: Sind die Daten gesendet worden, oder werden sie empfangen? Beides zugleich auf einem Draht gibt eine Massenkarambolage mit dem Erfolg, daß aus unseren gesendeten Daten ein Haufen Info-Schrott wird, der bestenfalls noch im Trashcan ein Plätzchen findet.
Bleibt also nur, den Sende- und Empfangsdaten getrennte Leitungen zu spendieren, macht mit GND (Bezugspunkt der Spannungen, auch Ground oder Masse genannt) drei Verbindungen. Das war aber noch nicht alles. Ist unser Amiga Empfänger, muß er beim Sender nachfragen, ob und wann Daten kommen, dies geht ebenfalls nicht über die Datenleitung (vier Drähte). Ist der Amiga Sender, muß er dem Empfänger mitteilen, daß jetzt 'Ohren auf' gilt, dies geht je doch wiederum nicht auf der 'Empfangsbereit'-Leitung; denn wenn Sender und Empfänger auf gleichem Anschluß gegensätzliche Ansichten äußern, kann ebenfalls nichts funktionieren (fünf Drähte).
Nun, um die ganze Sache abzukürzen: Ein serielles Kabel hat mehrere Verbindungsleitungen, um die Daten mit anderen Geräten auszutauschen. Um zwei Computer direkt miteinander zu verbinden, ist es nötig, die Empfangsleitung des einen mit der Sendeleitung des anderen und die Sendeleitung des einen mit der Empfangsleitung des anderen zu verbinden. Die Kontrolleitungen müssen jetzt noch so geschaltet werden, daß beide Rechner den jeweiligen Zustand selbst erkennen. Ein solches Kabel nennt man Nullmodem-Kabel, eben weil es ohne Umwandlung durch das Modem auskommt.
Die Pinbelegung des entsprechenden Kabels finden Sie in Abbildung 1. Dieses Kabel sollte eigentlich mit fast allen Computern funktionieren, die über eine RS232C-(V.24-) Schnittstelle verfügen. Bevor Sie jetzt lange suchen, Sie finden sie unter der Bezeichnung 'serial' auf der Rückseite Ihres Amiga 500, 1000 oder 2000. Beim Amiga 1000 gibt es gratis dazu noch ein Problem: Seine Schnittstelle entspricht in der Pinbelegung nicht der Norm (wenn man von einer Norm bei Pinbelegungen sprechen kann), die Signale liegen nämlich spiegelverkehrt an. Für den Anschluß des Nullmodem-Kabels ist also auch noch die intensive Suche nach dem richtigen Anschlußbild nötig. Sie haben übrigens richtig gelesen bei 'sollte funktionieren'. Leider meinen manche Computerhersteller mit 'Norm' ihre eigene, und so kann es schnell passieren, daß die 'kompatible Schnittstelle' eben nur angeblich kompatibel ist. Mit unserem Kabel hat es jedoch einwandfrei zwischen einem Amstrad PC 1512 und einem A2500, sowie einem Peacock AT und einem A500 funktioniert, so daß es fast auszuschließen ist, daß hier Probleme entstehen. Falls nicht, gibt's nur die Möglichkeit, zu experimentieren; eine denkbare Hilfe wäre zum Beispiel das Entfernen der Leitungsverbindungen 4 -> 5 und 5 <- 4, wobei die Pins 4 und 5 im Stecker verbunden werden müssen.
Soweit zum Lötteil, jetzt kommt endlich die Software.

Das Schönste, was man mit seriellen Schnittstellen machen kann, ist übertragen (O-Ton Kermit)

'Kermit' ist eine Wortzusammenstellung aus der keltischen Sprache und bedeutet sinngemäß 'Kostet nichts', 'umsonst' (vielleicht auch 'Public Domain'?). Ähnlichkeiten mit einem berühmten Fernsehstar sind also nicht festzustellen. Festzustellen ist aber eines: Kermit ist für sehr viele Computer vorhanden, darunter sogar für VAX- und UNIX-Anlagen im Großrechnerbereich. Interessant wären für unsere Belange die CP/M-2.0- oder CP/M-3.0-Versionen. CP/M ist ein Betriebssystem, was in diesen Versionen auf vielen 8-Bit-Computern zu finden ist, dazu gehören zum Beispiel die Amstrad CPCs und Joyce, der Commodore 128, in der CP/M86-Version auch für viele 'Ur'-PCs. Atari ST und MS-DOS gehören inzwischen auch zum Kermit-Umfeld, 'MSKERMIT' liegt schon in Versionen bis 2.3 vor (falls nicht noch höher). Damit ist ein Großteil der Amiga-Vorgänger und -Mitbewerber schon abgedeckt.
Sicher, für den Amiga gibt es inzwischen schon eine Menge neuer (und vielleicht auch besserer) Übertragungsprogramme, jedoch hat keines die Verbreitung erreicht wie Kermit. Ein Grund mehr also, sich das Programm zuzulegen. Da es außerdem Public Domain ist, besteht kein Grund, einen Kauf nicht zu riskieren.
Durch die weite Verbreitung von Kermit läßt sich auf verschiedenen Computern trotzdem eine Angleichung vornehmen, was bei unterschiedlichen Computertypen sonst kaum möglich ist.
Was kann nun Kermit? Kermit läßt sich von der Workbench wie auch von der Shell

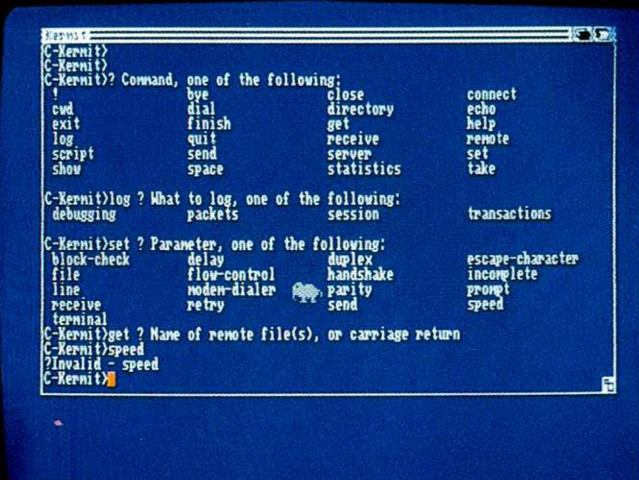


Abbildung 1. Die Befehle von Kermit. Ob der Rechner nun als Server, Sender oder Empfänger fungiert, sie haben alle Einstellungsmöglichkeiten.

Liste der Bauteile

2 SUB-D-Buchsen 25poliger Lötanschluß
2 Gehäuse für 25polige SUB-D
8poliges abgeschirmtes Kabel
in Rundausführung (bis zu 5 Meter)

Als Buchsengehäuse sind 'Posthauben' ideal (Metallausführung), die Abschirmung sollte elektrisch mit dem Metall verbunden werden.

(oder dem CLI) starten. Es bietet eine Vielzahl an Einstellmöglichkeiten. So läßt sich die Baudrate einstellen, dabei werden alle Werte akzeptiert. Sinnvoll ist es jedoch, erstens nur Standardwerte einzugeben und zweitens beide Computer, Sender und Empfänger, auf die gleiche Baudrate anzugleichen (Baud = Bit pro Sekunde). Kermit kann im Voll-Duplex-Betrieb arbeiten, das heißt, Daten können gleichzeitig empfangen und gesendet werden (deshalb existieren auch getrennte Sende- und Empfangsleitungen, ein Zusammenstoß ist damit ausgeschlossen).

Überhaupt ist Kermit sehr anwenderfreundlich. Durch betätigen der <SHIFT> + <?> -Tasten werden alle Kommandos aufgerufen. Gibt man das Kommando mit Fragezeichen ein, werden die erwarteten Parameter angezeigt. Fehler werden sofort reklamiert, führen also nicht zu Problemen. Baudraten, die nicht dem Standard entsprechen, werden zwar akzeptiert, der Benutzer wird jedoch darauf hingewiesen, daß die eben eingegebene Baudrate unüblich ist.

Kermit läßt das Einlesen des jeweiligen Disketteninhalts zu, DIR wird dazu wie im DOS benutzt. Die zu übertragenden Files sollten im ASCII-Code vorliegen. Steuerzeichen werden beim Protokoll vermerkt und angezeigt, aber trotzdem losgeschickt, was auf dem Empfänger recht eigenwillige Textpassagen erzeugen kann. Probleme beim Übertragen lassen sich übrigens jederzeit abrufen, unbekannte Phänomene fallen damit in die Kategorie 'vergessen'.

Interessant ist auch die Möglichkeit, Computer als 'Server' einzusetzen. Dazu ein einfaches Beispiel:

Durch SET BAUD 19200 auf beiden Rechnern die Baudrate setzen, dann CONNECT eingeben, ebenfalls auf beiden Computern – und schon kann man auf Tastatur 1 etwas eingeben, was gleichzeitig auf Monitor 2 zu sehen ist und umgekehrt.

Wie gesagt, Kermit enthält viele Funktionen, alle aufzuzählen, ist ein schwieriges Unterfangen. Nur soviel: Wer sich ein Nullmodem-Kabel baut (zu kaufen sind sie auch, mit zirka 20 bis 30 DM aber teurer) und sich die entsprechenden Kermit-Disketten zulegt, hat gute Chancen, seine alten Daten zu retten oder Arbeit von fremden Computern auf dem Amiga zu beenden, vorausgesetzt, der 'Ex' verfügt ebenfalls über eine RS-232C-Schnittstelle. Sicher, die A2000- (oder höher) Besitzer haben ihren PC-Teil mit AREAD und AWRITE, trotzdem bietet ein Übertragungsprogramm mit Kabel mehr Komfort. Diejenigen, die einen A500 oder A1000 besitzen, sind da noch besser bedient.

Ach übrigens – manche Spiele lassen einen Zwei-Spieler-Modus über die serielle Schnittstelle zu; was meinen Sie, wie interessant es sein kann, mit seinem Mitspieler über zwei Rechner zu kommunizieren. Auch der Flight-Simulator II bietet diese Möglichkeit an. Mal sehen, wer besser zum Ziel kommt. Überhaupt bekommen selbst Flugsimulationen wie 'F15 Strike Eagle' andere Dimensionen, wenn der Gegner plötzlich nicht mehr der Computer, sondern jemand aus der Familie ist und Sie gnadenlos per Kabel durch die imaginären Lüfte jagt. Selbst eingefleischte Pazifisten können da nicht widerstehen, ihren engsten Verwandten eine Rakete nachzuschicken.

(jb)

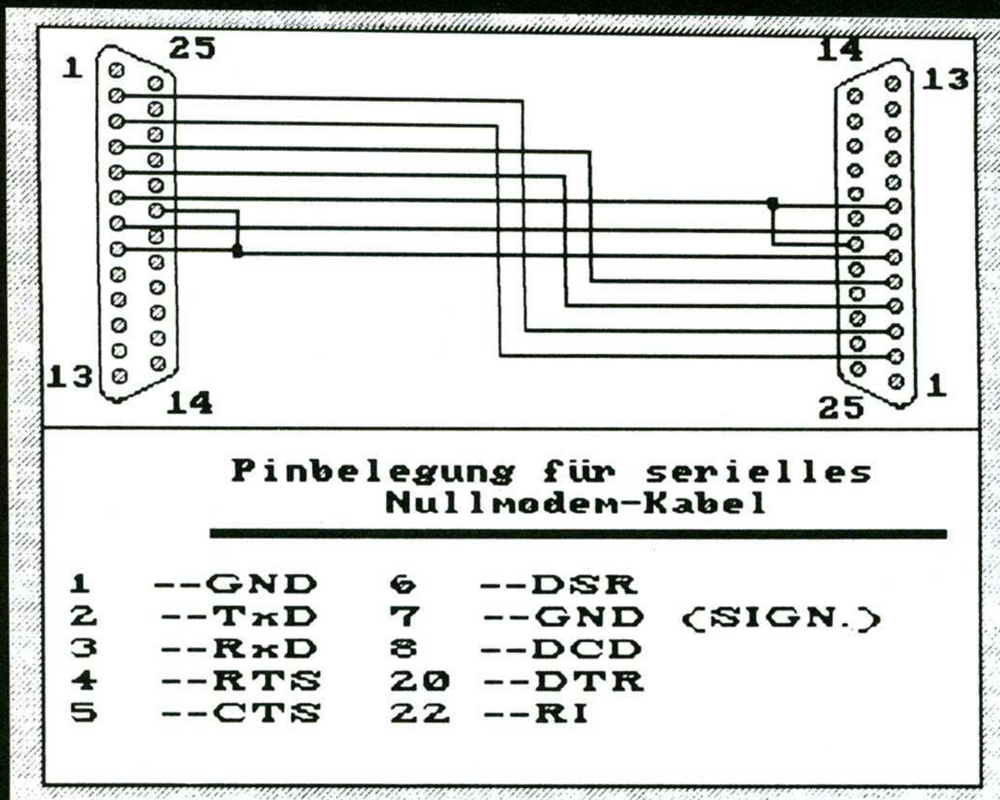


Abbildung 2. Die Zeichnung des seriellen Schnittstellenkabels. Benötigt werden nur wenige Bauteile.

Markus Steimar

Basteleien, nicht nur für Anfänger

Reset und Prozessorbremse für Amiga 500 und 1000 im Eigenbau

Hardwarebasteleien können Abenteuer werden. Jemand, der noch nie in seinem Leben einen Lötkolben in der Hand hatte, verspürt angesichts im Eigenbau aufgemotzter Hardware einen Hauch von Ehrfurcht gegenüber dem versierten Hardwarebastler. Allein wie richtig gelötet wird, wird für den Laien zum Problem. Was wirklich fehlt, ist ein vernünftiges Projekt und eine genaue Anleitung…

Wo fang ich an, fragt sich der Interessierte, überlegt und besorgt sich zunächst einmal einen Lötkolben. Nur welches Fabrikat? Welche Leistung? Das Angebot ist vielfältig.

Tatsächlich gibt es für unterschiedliche Aufgaben verschiedene Lötkolben. Der Anfänger sollte sich einen Kolben mit 25 Watt Leistung und einer Feinspitze zulegen. Alle Geräte mit höherer Leistung eignen nur zum Weißblechschneiden und noch rabiateren Anwendungen. Zu unserer Bastel-Grundausstattung gehören allerdings noch ein paar weitere Kleinigkeiten: Lötzinn und eine Ablötpumpe. Also besorgen wir uns Feinstlötzinn mit mindestens 60% Zinnanteil. Ergänzt wird das Ganze durch eine gute Schreibtischlampe (wir wollen sehen, was wir löten), eine saubere Arbeitsfläche und einen bequemen Stuhl. Nun ist die benötigte Ausrüstung beisammen; was uns fehlt, ist ein Projekt, das nicht zu umfangreich ist, aber trotzdem sinnvoll sein muß. (Ein wesentliches Motivationsmoment, oder geben Sie sich beim Produzieren von Unsinn Mühe?)

Stückliste für Resetschalter und Bremse am Amiga-Expansionsport

1 Stück Amiga-DMA-Portexpander,
1 Stück Kipphebel- oder Wippenschalter,
1 Stück Taster

Als sinnvolle kleine Hardwareprojekte würden sich ein Resetschalter und eine Ablaufbremse für unseren Amiga nicht schlecht machen. Tatsächlich ist beides recht einfach zu realisieren, es brauchen am Erweiterungsport des Amiga nur ein paar Drähte mit bestimmten Pins verlötet zu werden. Die Drähte werden mit einem Schalter verbunden, der bei Betätigen die entsprechende Funktion auslöst. Einzelheiten über die benötigten Bauteile entnehmen Sie bitte der Stückliste.

Nächster Schritt im Zuge unserer Hardwarebastelei ist das Studium des Bauplans. Hier gilt es, sich zunächst einmal zurechtzufinden. Der Amiga-Erweiterungsport besteht aus 85 Kontakten. Wo Kontakt eins und wo Kontakt 85 liegen, können Sie Ihrer Amiga-Dokumentation entnehmen (bitte achten Sie darauf, daß die Expansionsports des Amiga 500 und die des Amiga 1000 umgekehrt sind).

Wir könnten nun mit der Arbeit beginnen und unsere Drähte an die entsprechenden Kontakte löten. Nur werden Sie irgendwann einmal solch voreiliges Handeln verfluchen. Zum Beispiel dann, wenn Sie eine Speichererweiterung oder einen Festplattencontroller an den mit Drähten und Lötzinn verschönten Expansionsport anschließen wollen. Und damit haben wir auch schon eine der elementaren Regeln entdeckt, die bei Hardwarebasteleien dringendst beherzigt werden sollten. Zunächst sollte das ganze Projekt genau überdacht werden, insbesondere mögliche Schäden sollten vor Arbeitsbeginn ins Kalkül gezogen werden. Voreilige und Unbedachte haben schnell, sehr zur Freude der Hersteller, die teure Hardware ruiniert und müssen kostspielige Reparaturen in Kauf nehmen. Uns hilft ein DMA-Portexpander weiter. Diese Erweiterung wirkt im Grunde wie eine Weiche für den Port. An dem Portexpander befinden sich fünf, jeweils einzeln anschaltbare Ports. Wir können also unsere Lötarbeiten an einem dieser Ports vornehmen, ohne den Amiga zu gefährden. Ist alles fertig, muß der Portexpander nur noch an den Expansionsport des Amiga angeschlossen werden.

Doch zurück zu unserer eigentlichen Bastelei. Wie Sie auf den Bauplänen sehen, werden bei der Bremsenschaltung am Amiga 500 und am Amiga 1000 die Pins 1 und 3, beides Masseleitungen, miteinander verbunden. An Pin 3 wird nun ein Draht angelötet, der mit einem Kontakt des Schalters verbunden wird. Ebenso wird an Pin 55, der Halteleitung des Prozessors, ein weiterer Draht angelötet, dessen anderes Ende mit dem freien Kontakt des Schalters verbunden wird. Auch bei der Resetschaltung werden bei Amiga 500 und 1000 die Masseleitungen von Pin 1 und 3 miteinander verbunden. Wie schon bei der Bremse wird nun an Pin 3 ein Draht angelötet und dessen anderes Ende mit einem Kontakt des Tasters verbunden. Ein zweiter Draht wird an Pin 53 angelötet und mit dem anderen Kontakt des Tasters verbunden. Es ist durchaus möglich, beide Basteleien an einer Porterweiterung zu installieren. Es sollte nur dringend darauf geachtet werden, daß zwischen den Drähten, die an Pin 53 und 55 angeschlossen sind, kein Kontakt zustandekommen kann. Was die Pins 1 und 3 betrifft, ist es eigentlich egal, an welchem von beiden der Draht angeschlossen wird, da es sich bei beiden um Masseleitungen handelt.

(hs)

Bitte beachten Sie, daß bei unsachgemäßer Handhabung Ihres Computers ein eventuell noch bestehender Garantieanspruch verfallen kann. Für Schäden, die durch eventuelle Fehler in den abgedruckten Bauanleitungen entstehen, kann weder die Haftung noch eine juristische Verantwortung übernommen werden.

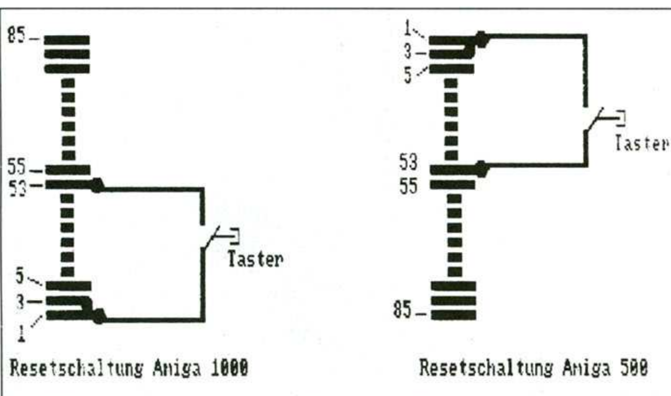


Bild 1. Bauplan für die Resetschaltung am Amiga 500 und 1000

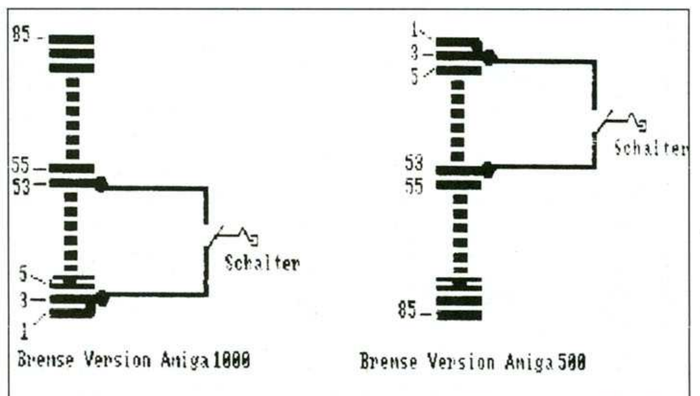
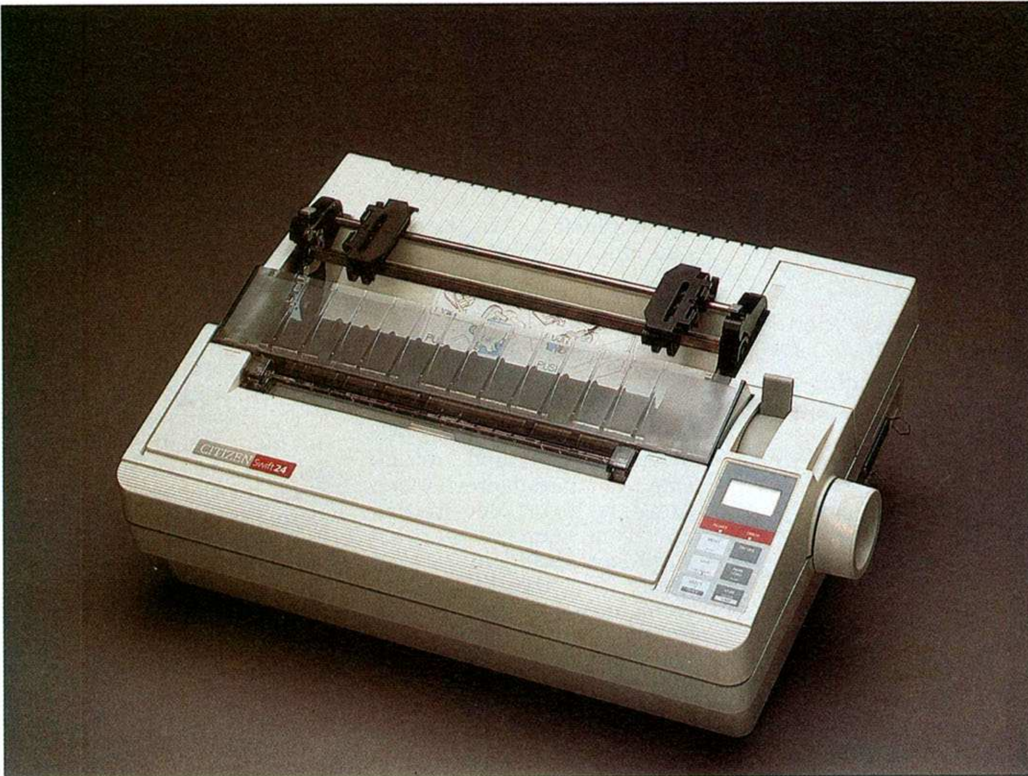


Bild 2. Bauplan für die Bremsenschaltung am Amiga 500 und 1000



Auf die Plätze, fertig, Druck

Citizen Swift 24 – ein Sprinter unter den Druckern

Beim Kauf eines Druckers hat der Käufer eines immer im Auge – "Value for money". Das heißt, er möchte für sein "sauer Erspartes" natürlich auch den entsprechenden Gegenwert haben. Die Firma Citizen hat mit dem Swift 24 einen 24-Nadel-Drucker auf den Markt gebracht, bei dem man wirklich "Value for money" bekommt.

Nehmen wir den Citizen Swift 24 etwas genauer unter die Lupe. Von den Ausmaßen her gehört dieser Drucker zu den kleineren. Auch sein äußeres Erscheinungsbild läßt sich am treffendsten mit handlich, kompakt beschreiben. Die Buchse für den Netzstecker liegt an der rückwärtigen Seite. Der parallele Anschluß führt über eine Buchse, die an der rechten Seite des Druckers angebracht ist. Oberhalb dieser Buchse besteht die Möglichkeit, den Drucker mit einer seriellen Schnittstelle zu versehen. Besondere Aufmerksamkeit erregt ein kleines Sensor-Tastenfeld mit LCD-Anzeige auf dem Drucker. Der Papiereinzug kann entweder von hinten oder aber von unten vollzogen werden. Dies ist besonders wichtig, wenn man Platzprobleme hat. Wie sieht nun das Innenleben des Swift 24 aus?

Außen hui und innen...?

Mit einem einzigen Handgriff läßt sich der Swift 24 aufklappen und zeigt sein Inneres. Ohne größere Probleme läßt sich das Farbband einsetzen. Der Druckkopf läßt sich ebenso leicht und schnell wechseln. Der Einzelblatteinzug erfolgt wahlweise manuell oder automatisch. Die Einzelblattführung läßt sich leicht aufsetzen, so daß man die einzelnen Blätter dann problemlos in den Schacht einführen kann. Auffällig beim Innenleben des Swift 24 ist ein kleines Fach an der rechten Seite des Druckers, wo sich ein Steckplatz für Schriftartkarten befindet, der dem Drucker eine größere Flexibilität und Vielseitigkeit verleiht.

Ein weiterer Punkt, der positiv zu bewerten ist, ist die Möglichkeit, Farboptionen zu wählen, um so in sieben verschiedenen Farben zu drucken. Man benötigt dazu ein mehrfarbiges Farbband und eine maschinelle Farbeinheit. Der temporäre Speicher des Druckers läßt sich mit einem RAM-Chip von den standardmäßigen 8 kByte auf 40 kByte erweitern. Diese Zusatzoption lohnt sich in erster

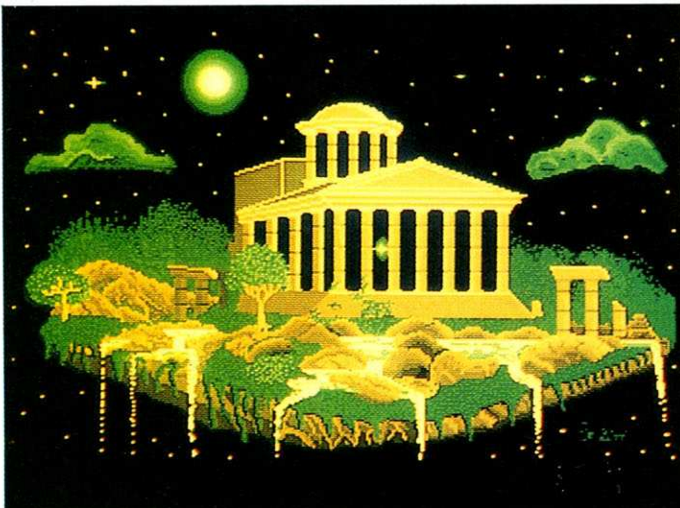


Bild 1. So sieht das Originalbild "Cloudnine" auf dem Amiga aus



Abbildung 1. Der Negativ-Ausdruck des 4096-Farben-Bildes "Cloudnine"

Linie dann, wenn größere Dokumentmengen verarbeitet werden sollen. Gehen wir noch etwas mehr auf die Leistungsfähigkeit dieses Druckers ein.

... ein leistungsfähiger Sprinter

Wie läuft die Konfiguration des Swift 24 ab? Hier tritt das Sensortastenfeld mit seiner LCD-Anzeige auf den Plan. Sechs verschiedene Tastenfelder machen die Konfiguration auch ohne Handbuch durch die gleichzeitige LCD-Anzeige fast zum Kinderspiel. So hat man immer den Überblick darüber, was gerade eingestellt wird und in welchem Untermenü man sich befindet. Weiterhin bestehen verschiedene Möglichkeiten zur Konfiguration: Zum einen kann von Menü zu Menü jeder Punkt einzeln angesprochen und ausgewählt werden. Schneller geht's jedoch mit dem Quick-Set-up, wo Makros angesprungen werden können; der Drucker bleibt dabei auf online geschaltet. Vier verschiedene Makros können dabei definiert werden, um die jeweilige Konfiguration darin zu speichern.

Eine Tastenfunktion fällt besonders ins Auge: die PARK-LOAD-Funktion. Sie dient dazu, die Seitenanfangsposition und den Mikro-Zeilenvorschub zu definieren und zu speichern. Ferner kann man mit dieser Funktion die Abreißposition sowie die horizontale Punktaussteuerung ganz genau bestimmen und natürlich auch speichern. Kommen wir zu einem weiteren sehr positiven Feature. Die Tatsache, daß Geschwindigkeit keine Hexerei ist, bewahrheitet sich beim Swift 24 auf sehr eindrucksvolle Weise. Der Probedruck der verschiedenen Schriftarten (Abbildung 1) erfolgte in "Null Komma nichts". Auch die Geschwindigkeit, mit der der Negativausdruck des 4096-Farben-Bildes "Cloudnine" (Abbildung 2) im LQ-Roman-Modus abgedruckt wurde, war ein überzeugender Beweis seiner Leistungsfähigkeit. Als Vergleich kann das Originalbild herangezogen werden (Bild 1). Der schnellste Modus ist der Draftmodus, auch Entwurfsmodus genannt. Hier werden die Zeichen mit einer Geschwindigkeit von 192 Zeichen pro Sekunde aufs Papier gebracht.

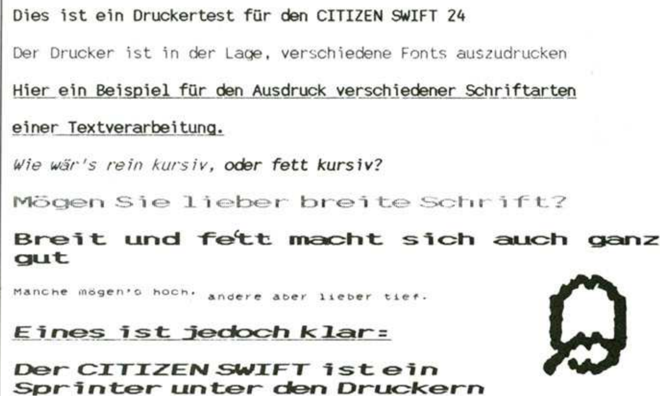
Insgesamt vier verschiedene Schriftarten stehen dem Benutzer zur Verfügung: Times Roman, Helvetica, Courier und Prestige. Mit weiteren Features, wie den Sonderschriftarten Outline, Shadow und Reverse, lassen sich Texte grafisch etwas aufpeppen. Die bereits erwähnten IC-Schriftkarten sind optional und gehören somit nicht zum Lieferumfang. Auffällig ist, daß der Citizen Swift 24 über unterschiedliche Emulationen verfügt. Der Swift kann dabei den Epson LQ 850, den IBM Proprinter und den NEC P6 emulieren. Sie können entweder per Tastenfeld eingestellt oder mit Hilfe von Escape-Sequenzen vom Computer an den Drucker geschickt werden.

Gesamteindruck positiv

Die Startposition des Druckkopfes wird mit einem Mikroschalter abgetastet und kann der Präzision des Druckwerks nichts anhaben. Beim Drucken selbst ist die Geräuschentwicklung sehr hoch, da das Gehäuse und das Druckwerk nicht schallgedämpft sind.

Im Bereich der 24-Nadler fällt der Citizen Swift 24 schon etwas aus dem Rahmen. Für einen Preis von 1098,– DM liefert er eine ganze Menge Quantität und auch Qualität, wo bei anderen Druckern dieser Preisklasse doch hier und da noch einiges im Argen liegt. Das einzige Manko dieses Druckers ist zugleich ein wenig unangenehm: Die Geräuschentwicklung ist relativ hoch. Hier könnten sich die Hersteller vielleicht noch etwas einfallen lassen. Ansonsten wird der sehr positive Eindruck, den der Swift 24 hinterläßt, in keinster Weise geschmälert.

(vb)


Dies ist ein Druckertest für den CITIZEN SWIFT 24

Der Drucker ist in der Lage, verschiedene Fonts auszudrucken

Hier ein Beispiel für den Ausdruck verschiedener Schriftarten einer Textverarbeitung.

*Wie wär's rein kursiv, **oder fett kursiv?***

Mögen Sie lieber breite Schrift?

Breit und fett macht sich auch ganz gut

Manche mögen's hoch. andere aber lieber tief.

Eines ist jedoch klar:

Der CITIZEN SWIFT ist ein Sprinter unter den Druckern

Abbildung 2. Eine Druckprobe des Swift 24, bei der verschiedene Schriftarten einer Textverarbeitung dargestellt werden

AMIGA DOS Blitzlicht

Name: Citizen Swift 24
Art: 24-Nadel-Matrixdrucker
Hersteller: Citizen Europe Deutschland 8056 Neufahrn
Vertrieb: Fachhandel
Preis: 1098,– DM

Positiv:
- sehr schnell
- verfügt über drei unterschiedliche Emulationen
- PARK-LOAD-Funktion
- Sonderschriftarten Outline, Shadow, Reverse
- leichte Konfigurierung durch Tastenfeld mit LCD-Anzeige
- Preis/Leistungsverhältnis
- optional: Farbdruck, Erweiterung des Druckpuffers auf 40 kByte, automatischer Einzelblatteinzug

Negativ:
- Geräuschentwicklung beim Drucken relativ hoch

Willi Bäcker

LinkChecker V2.0

Linkviren? – Mir doch egal!

Da es für einen Virus unmöglich sein sollte, ein Programm zu infizieren und dabei die Filelänge nicht zu verändern, könnte man doch auf den Gedanken kommen, ein Programm zu schreiben, das die tatsächliche Länge mit der in einer Batchdatei angegebenen Byte-Anzahl vergleicht. Somit dürften die gefährlichen Linkviren wohl keine Chance mehr haben.

Unser Programm LinkChecker V2.0 (im folgenden kurz LC genannt) soll diese Aufgabe übernehmen. Das Programm benötigt eine ASCII-Datei, in der die zu überprüfenden Programme und deren Programmlänge angegeben sein müssen. Abbildung 1 zeigt, wie diese Batchdatei aussehen könnte.

Da es mehrere ASCII-Files auf Diskette geben kann, muß sich unser Parameterfile von den anderen Textfiles unterscheiden. Dies geschieht durch den Befehl LINK, der am Anfang des Parameterfiles stehen muß. Dabei braucht auf Groß- oder Kleinschreibung nicht geachtet zu werden.

```
LINK

\df0:devs
mountlist                  75        ;Dies ist lediglich
DiskMaster.Config          706       ;ein Beispiel für
system-configuration       232       ;den Aufbau eines

\df0:libs
mathtrans.library          4280      ;Parameterfiles.
icon.library               5996
ConHandler.library         6152      ;Die hier angegebenen
translator.library         10592     ;File-Längen müssen
info.library               15744     ;nicht übereinstimmen!
mathieeedoubbas.library    4352
version.library            400
diskfont.library           3968


\df0:
LinkChecker                2588

\df0:c
run                        2324
setmap                     4500
install                    1800
Delete                     5568
Type                       2112
FailAt                     1072
copy                       6684
list                       8972
cd                         1908
```

Abb. 1. So könnte ein Parameterfile für den LinkChecker aussehen

Am Anfang stand LIST

Will man beispielsweise den Amiga-DOS-Befehl 'List' aus dem Verzeichnis df0:c (Originallänge 8972 Bytes) vergleichen lassen, sollte man sich an folgende Syntax im Parameterfile halten:

df0:c/list 8972

Also zuerst den Filename der zu überprüfenden Datei, dann die Bytelänge. Will man mehrere oder vielleicht alle Files des Directories df0:c testen, kann man im Parameterfile folgendes anwenden:

\df0:c
list 8972
run 2324
install 1800
delete 5568
type 2112
cd 1908

Zuerst wird überprüft, ob das Verzeichnis auch wirklich existiert. Wenn nicht, passiert bis zum nächsten Directorywechsel gar nichts mehr, das heißt, alle folgenden Files werden einfach ignoriert. Sollte es dieses Directory wider Erwarten doch geben, wird dort nach den aufgeführten Files gesucht. Man spart sich dadurch eine ganze Menge Tipparbeit.

Bei Bedarf können in dem Parameterfile auch Kommentare gesetzt werden, wie das folgende Beispiel zeigt:

\df0:c ; jetzt hab ich das Directory gewechselt
list 8972 ; list hat 8972 Bytes
\df0: ; jetzt bin ich wieder imStammverzeichnis

Den letzten Directorywechsel im obigen Beispiel kann man sich natürlich sparen, denn sobald das komplette Parameterfile abgearbeitet ist, wird sofort wieder in das Verzeichnis gesprungen, in dem man sich ursprünglich befand.

Die Zeilen **müssen** immer mit RETURN abgeschlossen werden. Außer diesem RETURN wird jeder Fehler im Parameterfile übrigens dann bei der Ausführung gnadenlos angezeigt. Aufrufen kann man den LC mit:

LinkChecker parameterfile

Hierbei wird nur eine kleine Message ausgegeben, und nur eventuell auftretende Abweichungen von Dateilängen werden angezeigt.

Man kann den LC aber auch mit verschiedenen Parametern aufrufen:

LinkChecker parameterfile -s

Dabei steht der Parameter -s für show, man wird also über jede Aktion informiert. Ein weiterer Parameter liegt mit dem Zusatz -v vor. Gibt man diese Option mit an (v steht für Virus), so werden nun KickTagPtr, KickMemPtr, CoolCapture, ColdCapture und der OpenLibrary-Aufruf der exec.library überprüft. Beide Optionen lassen sich auch verknüpfen. Der Aufruf wäre dann folgendermaßen:

LinkChecker parameterfile -sv

Wir brauchen sicherlich nicht zu erwähnen, daß unser Programm auch mit Fesplatten arbeitet – sonst wäre die Geschichte ja auch nur halb so interessant.

Sollte es tatsächlich vorkommen, daß der LC die Meldung "Filelength incorrect!" ausgibt, kann es durchaus sein, daß sich ein Linkvirus in diesem File breitgemacht hat. In diesem Fall wird beim Verlassen des LC der Return-Code (Register d0) auf den Wert Zehn gesetzt. Dies führt zum Abbruch in der Startup-Sequenz. Mit dem Befehl FAILAT 20 kann man diesen Abbruch jedoch übergehen. Das entsprechende File sollte man dann etwas genauer unter die Lupe nehmen.

(br)

Listings

```
;****************************************************
;*                 LinkChecker V2.0                 *
;*             written by Willi Bäcker              *
;*             (c) 1990 by DMV-Verlag               *
;*            Sprache: Assembler (Seka)             *
;****************************************************

Open                =         -30
Close               =         -30 -6
Read                =         -30 -12
Write               =         -30 -18
Output              =         -30 -30
Seek                =         -30 -36
Lock                =         -30 -54
UnLock              =         -30 -60
CurrentDir          =         -30 -96
Examine             =         -30 -72
```

Listing: LinkChecker V2.0

```

OpenLibrary         =         -30 -522
CloseLibrary        =         -30 -384
AllocMem            =         -30 -168
FreeMem             =         -30 -180
FindTask            =         -30 -264

EXEC:               equ       4
CSI:                equ       $9b

start:
                    lea       Path(pc),a1
                    andi.l    #$ff,d0
                    subq.l    #1,d0
                    move.l    d0,Paralength
                    beq       NoPara          ;there is no Parameter
CoPara:             cmp.b     #$0a,(a0)
                    beq.s     CReady
                    move.b    (a0)+,(a1)+
```

Listing: LinkChecker V2.0

```
                dbra    d0,CoPara
CReady:         lea     path(pc),a0
                cmp.w   #$3f00,(a0)     ;is it only a ?
                beq     abouthelp
                lea     path(pc),a0
                cmpi.l  #"info",(a0)
                beq     printinfos
                lea     path(pc),a0
OptLoop0:       cmpi.b  #" ",(a0)+
                bne.s   OptLoop0
                clr.b   -1(a0)
                cmpi.b  #"-",(a0)+
                bne     illegaloption
OptLoop1:       cmpi.b  #0,(a0)
                beq.s   optout
                cmpi.b  #"s",(a0)
                beq.s   foundS
                cmpi.b  #"v",(a0)
                beq.s   foundV
                bra     illegaloption
founds:         st      OptionFlag
                tst.b   (a0)+
                bra.s   OptLoop1
foundv:         st      OptionFlag1
                tst.b   (a0)+
                bra.s   OptLoop1
optout:         move.l  EXEC,a0
                move.l  $2a(a0),ColdCapture
                move.l  $2e(a0),CoolCapture

                move.l  $222(a0),KickMemPtr
                move.l  $226(a0),KickTagPtr
                move.l  -$196(a0),OpenPtr
                bsr     OpenLib
                tst.b   OptionFlag
                beq.s   no_clear
                move.l  #ClearText,d0
                bsr     DosPrint
No_Clear:       tst.b   OptionFlag1
                beq.s   Vir5
                tst.l   ColdCapture
                beq.s   Vir1
                move.l  #ColdCaptureText,d0
                bsr     DosPrint
Vir1:           tst.l   CoolCapture
                beq.s   Vir2
                move.l  #CoolCaptureText,d0
                bsr     DosPrint
Vir2:           tst.l   KickMemPtr
                beq.s   Vir3
                move.l  #KickMemPtrText,d0
                bsr     DosPrint
Vir3:           tst.l   KickTagPtr
                beq.s   Vir4
                move.l  #KickTagPtrText,d0
                bsr     DosPrint
Vir4:           move.l  OpenPtr,d0
                swap    d0
                cmpi.b  #$fc,d0
                beq.s   Vir5
                move.l  #OpenPtrText,d0
                bsr     DosPrint
Vir5:           move.l  DosBase(pc),a6
                move.l  #Path,d1
                move.l  #1005,d2
                jsr     Open(a6)        ;Open
                tst.l   d0
                beq     NoParaFile
                move.l  d0,ParaHandle
                move.l  d0,d1
                move.l  #TestBuffer,d2
                moveq   #4,d3
                jsr     Read(a6)
                lea     TestBuffer(pc),a0
                moveq   #3,d0
test1:          cmp.b   #"a",(a0)
                bhi.s   test11
                add.b   #$20,(a0)+
test11:         dbra    d0,test1
                lea     Testbuffer(pc),a0
                cmp.l   #"link",(a0)    ;Test File
                bne     IncorrectFile
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                jsr     Close(a6)
                clr.l   ParaHandle
                move.l  DosBase(pc),a6
                move.l  #Path,d1
                move.l  #1005,d2
                jsr     Open(a6)        ;Open
                move.l  d0,ParaHandle
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                moveq   #0,d2
                moveq   #1,d3
                jsr     Seek(a6)

                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                moveq   #0,d2
                moveq   #0,d3
                jsr     Seek(a6)
                move.l  d0,ParaFilelength
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                moveq   #0,d2
                moveq   #-1,d3
                jsr     Seek(a6)
                move.l  EXEC,a6
                move.l  parafilelength(pc),d0
                addq    #8,d0
                move.l  #$10002,d1
                jsr     AllocMem(a6)
                tst.l   d0
                beq     MemoryError
                move.l  d0,memad0
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                move.l  memad0,d2
                move.l  ParaFileLength,d3
                jsr     Read(a6)
```

Listing: LinkChecker V2.0

```
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                jsr     Close(a6)
                move.l  #begintext,d0
                bsr     DosPrint
                tst.b   OptionFlag
                beq     noinfo
                move.l  #ParaText,d0
                bsr     DosPrint
                move.l  #Path,d0
                bsr     DosPrint
                move.l  #RetText,d0
                bsr     DosPrint
noinfo:         move.l  memad0(pc),a0   ;get TextBuffer
                addq.l  #4,a0
;---------------------------------------------------------------
doJob:          move.l  mytask(pc),a3
                btst    #12,28(a3)
                beq.s   huch
                move.l  #Exittext,d0
                bsr     DosPrint
                bra     Closeall
huch:           cmp.b   #0,(a0)         ;EndMark
                beq     Closeall
                cmp.b   #"!",(a0)+
                blo.s   doJob
                tst.b   -(a0)
                lea     NameBuffer(pc),a1
dojob1:         cmp.b   #"\",(a0)       ;followed by a D
ir Name
                beq     Directory
                cmp.b   #";",(a0)
                beq     Comment
                move.b  (a0)+,(a1)+
                cmp.b   #"!",(a0)
                bhs.s   dojob1
                lea     ZiffBuffer(pc),a1
dojob2:         cmp.b   #"!",(a0)+
                blo.s   doJob2
                tst.b   -(a0)
dojob3:         move.b  (a0)+,(a1)+
                cmp.b   #"!",(a0)

                bhs.s   dojob3
                move.l  a0,store0
                tst.b   DirFlag
                bmi     dojob
                lea     ZiffBuffer(pc),a4
                bsr     dec2bin         ;convert to hex
                tst.b   ZiffFlag
                bne     ValueError
                move.l  d0,Filelength
                tst.b   OptionFlag
                beq.s   nooptions
                movem.l d0-d7/a0-a6,-(sp)
                move.l  #workText,d0
                bsr     DosPrint
                move.l  #NameBuffer,d0
                bsr     DosPrint
                movem.l (sp)+,d0-d7/a0-a6
nooptions:
                move.l  DosBase(pc),a6
                move.l  #NameBuffer,d1
                move.l  #1005,d2
                jsr     Open(a6)
                tst.l   d0
                beq     Jobnotfound
                move.l  d0,Filehandle
                move.l  DosBase(pc),a6
                move.l  FileHandle(pc),d1
                moveq   #0,d2
                moveq   #1,d3
                jsr     Seek(a6)
                move.l  DosBase(pc),a6
                move.l  FileHandle(pc),d1
                moveq   #0,d2
                moveq   #0,d3
                jsr     Seek(a6)
                move.l  d0,Filelength1
                move.l  DosBase(pc),a6
                move.l  FileHandle(pc),d1
                moveq   #0,d2
                moveq   #-1,d3
                jsr     Seek(a6)
                move.l  DosBase(pc),a6
                move.l  FileHandle(pc),d1
                jsr     Close(a6)
                move.l  filelength(pc),d0
                move.l  filelength1(pc),d1
                cmp.l   d0,d1
                bne     notcorrect
clear:          lea     namebuffer(pc),a0
                moveq   #89,d0
nfl1:           clr.b   (a0)+
                dbra    d0,nfl1
                move.l  store0,a0
                bra     doJob
;---------------------------------------------------------------
Closeall:       tst.l   locker
                beq.s   nolocker
                move.l  locker,d1
                jsr     CurrentDir(a6)
nolocker:       tst.l   OldLock
                beq.s   nooldlock
                move.l  OldLock,d1
                jsr     CurrentDir(a6)
nooldlock:      move.l  #RetText,d0
                bsr     DosPrint
                move.l  EXEC,a6

                move.l  memad0(pc),a1
                move.l  ParaFileLength(pc),d0
                addq    #8,d0
                jsr     FreeMem(a6)
                move.l  #Cursor,d0
                bsr     DosPrint
                bsr     CloseLib
                moveq   #0,d0
                move.b  CLIFlag,d0
                sub.l   a0,a0
                rts
```

Listing: LinkChecker V2.0

```
;-----------------------------------------------------------
comment:        tst.b   (a0)+
comm1:          cmp.b   #10,(a0)+
                bne.s   comm1
                bra     dojob
;-----------------------------------------------------------
Directory:      bsr     cleardirbuffer
                lea     dirbuffer(pc),a1
                tst.b   (a0)+
dir0:           move.b  (a0)+,(a1)+
                cmp.b   #"!",(a0)
                bhs.s   dir0
                movem.l d0-d7/a0-a6,-(sp)
                tst.l   OldLock
                beq.s   no_oldlocka
                move.l  OldLock,d1
                move.l  DosBase,a6
                jsr     CurrentDir(a6)
no_oldlocka:    move.l  DosBase,a6
                move.l  #DirBuffer,d1
                moveq   #-2,d2
                jsr     Lock(a6)
                tst.l   d0
                beq     dirnotfound
                move.l  d0,locker       ;neuen DirLock h
olen
                move.l  d0,d1
                jsr     CurrentDir(a6)
                move.l  d0,OldLock      ;alten Lock spei
chern
                movem.l (sp)+,d0-d7/a0-a6
                tst.b   OptionFlag
                beq     noDiroption
                move.l  #worktext1,d0
                bsr     DosPrint
                move.l  #DirBuffer,d0
                bsr     DosPrint
                move.l  #RetText,d0
                bsr     DosPrint
noDirOption:    clr.b   DirFlag
                lea     NameBuffer(pc),a4
                moveq   #89,d0
dir5:           clr.b   (a4)+
                dbra    d0,dir5
                bra     dojob
;-----------------------------------------------------------
illegaloption:  bsr     OpenLib
                move.l  #Optiontext,d0
                bsr     DosPrint
                bsr     CloseLib
                moveq   #0,d0
                sub.l   a0,a0
                rts
;-----------------------------------------------------------
dirnotfound:    st      DirFlag

                tst.b   OptionFlag
                beq.s   nodir7
                move.l  #retText,d0
                bsr     DosPrint
nodir7:         move.l  #NoDirText,d0
                bsr     DosPrint
                move.l  #DirBuffer,d0
                bsr     DosPrint
                move.l  #asciitext2,d0
                bsr     DosPrint
                tst.b   OptionFlag
                bne.s   nodir3
                move.l  #RetText,d0
                bsr     DosPrint
nodir3:         movem.l (sp)+,d0-d7/a0-a6
                bra     DoJob
;-----------------------------------------------------------
cleardirbuffer: movem.l d0/a0,-(sp)
                lea     dirbuffer(pc),a0
                moveq   #99,d0
clearloop:      clr.b   (a0)+
                dbra    d0,clearloop
                movem.l (sp)+,d0/a0
                rts
;-----------------------------------------------------------
ValueError:     tst.b   OptionFlag
                beq.s   Value1
                move.l  #RetText,d0
                bsr     DosPrint
Value1:         move.l  #AsciiText,d0
                bsr     DosPrint
                move.l  #Namebuffer,d0
                bsr     DosPrint
                move.l  #AsciiText1,d0
                bsr     DosPrint
tt:             tst.b   OptionFlag
                bne.s   Value2
                move.l  #RetText,d0
                bsr     DosPrint
value2:         lea     NameBuffer(pc),a4
                moveq   #89,d0
vloop:          clr.b   (a4)+
                dbra    d0,vloop
                clr.b   ZiffFlag
                bra     dojob
;-----------------------------------------------------------
Jobnotfound:    tst.b   OptionFlag
                bne.s   job1
                move.l  #NameBuffer,d0
                bsr     DosPrint
job1:           move.l  #Filenfound,d0
                bsr     DosPrint
                tst.b   OptionFlag
                bne.s   job2
                move.l  #RetText,d0
                bsr     DosPrint
job2:           lea     namebuffer(pc),a0
                moveq   #89,d0
nf1:            clr.b   (a0)+
                dbra    d0,nf1
                move.l  store0,a0
                move.b  #5,CLIFlag
                bra     dojob
;-----------------------------------------------------------
notcorrect:     tst.b   OptionFlag
                bne.s   not2
```

Listing: LinkChecker V2.0

```
                move.l  #NameBuffer,d0
                bsr     DosPrint
not2:           move.l  #inctext,d0
                bsr     DosPrint
                move.l  #RetText,d0
                bsr     DosPrint
not4:           lea     namebuffer(pc),a0
                moveq   #89,d0
nfl2:           clr.b   (a0)+
                dbra    d0,nfl2
                move.l  store0,a0
                move.b  #10,CLIFlag
                bra     dojob
;-----------------------------------------------------------
NoPara:         bsr     OpenLib         ;DosLib oeffnen
                move.l  #NoParaText,d0
                bsr     DosPrint
                bsr     CloseLib        ;DosLib schliess
en
                moveq   #0,d0
                sub.l   a0,a0
                rts
;-----------------------------------------------------------
NoParaFile:     move.l  #NoParaFText,d0
                bsr     DosPrint
                move.l  #Path,d0
                bsr     DosPrint
                move.l  #CheckText,d0
                bsr     DosPrint
                bsr     CloseLib        ;DosLib schliess
en
                moveq   #0,d0
                sub.l   a0,a0
                rts
;-----------------------------------------------------------
IncorrectFile:  move.l  #IncorrectText,d0
                bsr     DosPrint
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                jsr     Close(a6)       ;CloseFile
                bsr     CloseLib        ;DosLib schliess
en
                moveq   #0,d0
                sub.l   a0,a0
                rts
;-----------------------------------------------------------
MemoryError:    move.l  #MemErrorText,d0
                bsr     DosPrint
                move.l  DosBase(pc),a6
                move.l  ParaHandle(pc),d1
                jsr     Close(a6)
                bsr     CloseLib
                moveq   #0,d0
                sub.l   a0,a0
                rts
;-----------------------------------------------------------
LibError:       moveq   #4,d1
Lib1:           moveq   #-1,d0
Lib2:           move.w  d0,d2
                andi.l  #$0f00,d2
                move.w  d2,$dff180
                dbra    d0,Lib2
                dbra    d1,lib1
                moveq   #1,d0
                sub.l   a0,a0
                rts

;-----------------------------------------------------------
OpenLib:        move.l  EXEC,a6
                sub.l   a1,a1
                jsr     FindTask(a6)
                move.l  d0,mytask
                lea     DosName(pc),a1
                moveq   #0,d0
                jsr     OpenLibrary(a6)    ;OpenLibrary
                beq.s   LibError
                move.l  d0,DosBase      ;store DosBase
                move.l  d0,a6
                jsr     Output(a6)
                move.l  d0,CliHandle
                rts
;-----------------------------------------------------------
CloseLib:       move.l  EXEC,a6
                move.l  DosBase(pc),a1
                jsr     CloseLibrary(a6)
                rts
;-----------------------------------------------------------
DosPrint:       movem.l d0-d3/a0/a6,-(sp)  ;alle Regist
er auf STACK
                move.l  d0,d2           ;start retten
                move.l  d2,d3
                move.l  d2,a0           ;Stringadresse u
ebergeben
DosPrint0:      tst.b   (a0)+           ;Byte testen & A
dresse + 1
                bne.s   DosPrint0       ;Sprung wenn nic
ht null
                sub.l   a0,d3           ;
                neg.l   d3              ;positiv machen.
                subq    #1,d3
                move.l  Clihandle,d1
                move.l  DosBase,a6      ;Basisadr. holen
                jsr     Write(a6)       ;(dos.library) =
-48(a6)
                movem.l (sp)+,d0-d3/a0/a6  ;Register vo
m STACK holen
                rts                     ;zurueck
;-----------------------------------------------------------
abouthelp:      bsr     openlib
                move.l  DosBase(pc),a6
                move.l  #AboutText,d0
                bsr     DosPrint
                bsr     closelib
                moveq   #0,d0
                rts
;-----------------------------------------------------------
PrintInfos:     bsr     openlib
                move.l  DosBase(pc),a6
                move.l  #InfoText,d0
                bsr     DosPrint
                bsr     closelib
                moveq   #0,d0
                rts
```

Listing: LinkChecker V2.0

```
;--------------------------------------------------------
;Dezimalstring to Binaer
; a4 = Zeiger auf String
; d0 = Ergebnisregister
Dec2Bin:movem.l d1-d7/a0-a6,-(sp)
                moveq    #0,d0          ;Ergebnisregister
                moveq    #0,d2          ;Naechste Ziffer
DecLoop:        move.b   (a4)+,d2       ;d2 = Zeichen aus
String
                tst.b    d2

                beq.s    decout
                cmpi.b   #'0',d2        ;Dezimalziffer?
                blo.s    DecOut1
                cmpi.b   #'9',d2
                bhi.s    DecOut1        ;nein -->
                add.l    d0,d0
                move.l   d0,d1
                add.l    d0,d0
                add.l    d0,d0
                add.l    d1,d0
                subi.b   #$30,d2
                add.l    d2,d0
                bra.s    DecLoop
DecOut:         subq.l   #1,a4
                movem.l (sp)+,d1-d7/a0-a6
                rts
DecOut1:        st       ZiffFlag
                bra.s    DecOut
;--------------------------------------------------------
;Binaerzahl in Dezimal String wandeln
;a5= Zeiger auf StringBuffer
;Zahl steht in d4
BinDez:         movem.l d0-d7/a0-a6,-(sp)
                tst.l    d4              ;Zahl in d4
                beq.s    BinDez6         ;branch if null
                bmi.s    BinDez1         ;branch if Minus
                neg.l    d4              ;neg it
                bra.s    BinDez2
BinDez1:        move.b   #'-',(a5)+
BinDez2:        lea      Decathlon,a0
                clr.w    d1
BinDez3:        move.l   (a0)+,d2
                beq.s    BinDez6
                moveq    #-1,d0
BinDez4:        add.l    d2,d4
                dbgt     d0,BinDez4
                sub.l    d2,d4
                addq.w   #1,d0
                bne.s    BinDez5
                tst.w    d1
                beq.s    BinDez3
BinDez5:        moveq    #-1,d1
                neg.b    d0
                addi.b   #$30,d0
                move.b   d0,(a5)+
                bra.s    BinDez3
BinDez6:        neg.b    d4
                addi.b   #$30,d4
                move.b   d4,(a5)+
                movem.l (sp)+,d0-d7/a0-a6
                rts
Decathlon:
                dc.l     1000000000
                dc.l     100000000
                dc.l     10000000
                dc.l     1000000
                dc.l     100000
                dc.l     10000
                dc.l     1000
                dc.l     100
                dc.l     10
                dc.l     0
;--------------------------------------------------------
;--------------------------------------------------------
DosName:        dc.b     "dos.library",0
ColdCapture:    dc.l     0

CoolCapture:    dc.l     0
KickMemPtr:     dc.l     0
KickTagPtr:     dc.l     0
OpenPtr:        dc.l     0
DosBase:        dc.l     0
CliHandle:      dc.l     0
store0:         dc.l     0
store1:         dc.l     0
memad0:         dc.l     0
ParaLength:     dc.l     0
ParaFileLength: dc.l     0
ParaHandle:     dc.l     0
OldLock:        dc.l     0
Locker:         dc.l     0
infolock:       dc.l     0
Path:           blk.b    80,0
NameBuffer:     blk.b    80,0
ZiffBuffer:     blk.b    10,0
TestBuffer:     dc.l     0
Filelength:     dc.l     0               ;converted by AS
CIItable
Filelength1:    dc.l     0               ;original length
FileHandle:     dc.l     0
mytask:         dc.l     0
ZiffFlag:       dc.b     0,0
CLIFlag:        dc.b     0,0
OptionFlag:     dc.b     0,0
OptionFlag1:    dc.b     0,0
DirFlag:        dc.b     0,0
DirBuffer:      blk.b    100,0
align    4
infoBlock:      blk.b    280,0
;--------------------------------------------------------
;Texts:
ReturnText:     dc.b     10
RetText:        dc.b     10,0
ClearText:      dc.b     $c,0
NoParaText:     dc.b     10,10
                dc.b     "ERROR !",10
                dc.b     "USAGE: LinkChecker [FileName]"
                dc.b     10,10,0
;--------------------------------------------------------
NoParaFText:    dc.b     10,10
                dc.b     "ERROR !",10
```

Listing: LinkChecker V2.0

```
                dc.b     "Cannot find parameterfile : "
                dc.b     0
;--------------------------------------------------------
MemErrorText:   dc.b     10,10
                dc.b     "ERROR !",10
                dc.b     "Cannot allocate needed memorys
ize.",10
                dc.b     "Reset your AMIGA and try again
!",10
                dc.b     0
;--------------------------------------------------------
IncorrectText:  dc.b     10,10
                dc.b     "ERROR !",10
                dc.b     "This is not a correct paramete
r file !",10
                dc.b     "Be sure that your parameter fi
le "
                dc.b     "starts with 'LINK' !",10
                dc.b     10,0
;--------------------------------------------------------
AboutText:
                dc.b     10,10,10

                dc.b     "USAGE :",10
                dc.b     "LinkCheckerV2.0    [File"
                dc.b     "name]  Options  :  [-sv]",10
                dc.b     "[v] = Virus+pointer chec"
                dc.b     "k   [s] = visual control",10
                dc.b     10,0
;--------------------------------------------------------
inctext:        dc.b     $d,CSI,"54C",CSI,"7;31;40m"
                dc.b     "Filelength incorrect !"
                dc.b     CSI,"0;31;40m"
                dc.b     0
;--------------------------------------------------------
FilenFound:     dc.b     $d,CSI,"54C",CSI,"7;31;40m"
                dc.b     "File not found error !"
                dc.b     CSI,"0;31;40m"
                dc.b     0
;--------------------------------------------------------
NoDirText:      dc.b     CSI,"7;31;40m"
                dc.b     "Directory  not found >"
                dc.b     CSI,"0;31;40m ",0
;--------------------------------------------------------
AsciiText:      dc.b     CSI,"7;31;40m"
                dc.b     "Illegal parameter at >"
                dc.b     CSI,"0;31;40m ",0
;--------------------------------------------------------
AsciiText1:     dc.b     $d,CSI,"54C",CSI,"7;31;40m"
                dc.b     "Only DECIMAL numbers !"
                dc.b     CSI,"0;31;40m ",0
;--------------------------------------------------------
AsciiText2:     dc.b     $d,CSI,"54C",CSI,"7;31;40m"
                dc.b     "Check  parameterfile !"
                dc.b     CSI,"0;31;40m ",0
;--------------------------------------------------------
CheckText:      dc.b     10,"Check: Drive/Path/Filename.
",10,0
;--------------------------------------------------------
OptionText:     dc.b     10
                dc.b     "ERROR !!  Illegal option recog
nized."
                dc.b     10,10,0
;--------------------------------------------------------
workText:       dc.b     $0d,CSI,$4d
                dc.b     "processing..........> "
                dc.b     0
;--------------------------------------------------------
worktext1:      dc.b     10,CSI,$4d
                dc.b     "Directory changed to > "
                dc.b     0
;--------------------------------------------------------
begintext:      dc.b     10,CSI,$30,$20,$70
                dc.b     "Linkchecker V2.0 running.",10,
0
;--------------------------------------------------------
ParaText:       dc.b     "Parameterfile: > ",0
;--------------------------------------------------------
ExitText:       dc.b     "^C",10
;--------------------------------------------------------
Cursor:         dc.b     CSI,$20,$70,0
;--------------------------------------------------------
InfoText:       dc.b     10,10
                dc.b     '******************************
**',10
                dc.b     '        LinkChecker V2.0',10
                dc.b     '     written by Willi Baecker',
10
                dc.b     '   a FACTOR 5 production 1989'

,10
                dc.b     '******************************
**',10
                dc.b     0
ColdCaptureText:
                dc.b     'WARNING !!! ColdCapture is pat
ched !!'
                dc.b     '  Maybe a VIRUS in your Amiga
!',10,0
CoolCaptureText:
                dc.b     'WARNING !!! CoolCapture is pat
ched !!'
                dc.b     '  Maybe a VIRUS in your Amiga
!',10,0
KickMemPtrText:
                dc.b     'WARNING !!! KickMemPointer is
patched !!'
                dc.b     '  Maybe a VIRUS in your Amiga
!',10,0
KickTagPtrText:
                dc.b     'WARNING !!! KickTagPointer is
patched !!'
                dc.b     '  Maybe a VIRUS in your Amiga
!',10,0
OpenPtrText:
                dc.b     'WARNING !!! OpenLibrary is pat
ched !!'
                dc.b     '  Maybe a LINK-VIRUS in your A
miga !',10,0

                align    4
;--------------------------------------------------------
;--------------------------------------------------------
```

Listing: LinkChecker V2.0

Willi Bäcker

LinkList

Das Utility zum LinkChecker

Es ist sicherlich etwas umständlich im Umgang mit dem LinkChecker, sich alle Files, die überprüft werden sollen, zuerst listen zu lassen und dann mit der jeweiligen Programmlänge in ein Parameterfile zu bringen. Das Programm LinkList soll hier Abhilfe schaffen.

Das Programm LinkList gibt nach dem Aufruf alle Files eines Directories aus. Zusätzlich wird noch die Länge des Files und der Directoryname angegeben.
Die Hauptaufgabe von LinkList besteht darin, die Parameterfiles für den LinkChecker V2.0 einfach und schnell zu erweitern.

Es gibt mehrere Möglichkeiten, das Programm aufzurufen:

1.) LinkList df0:
Hier wird zuerst der Directoryname auf dem Bildschirm ausgegeben (also \df0:), gefolgt von den Files, die im Stammverzeichnis von df0: enthalten sind.

2.) LinkList >ram:test df0:
Wie bei Punkt 1, nur mit dem Unterschied, daß die Ausgabe nicht mehr auf den Bildschirm erfolgt, sondern in die Datei "ram:test" umgeleitet wird. Diese Datei kann mit einem beliebigen Editor verändert und für den LinkChecker bearbeitet werden.

3.) LinkList df0:c df1:devs/ParameterFile
Der dritte Aufruf dieses Programms unterscheidet sich generell von den beiden anderen. In diesem Fall wird zuerst versucht, eine bereits vorhandene Parameterliste (hier df1:devs/ParameterFile) in den Speicher zu laden. Sollten dabei Probleme auftreten wie: 'File nicht vorhanden' oder 'Kein Parameterfile' (aller Wahrscheinlichkeit nach fehlt die Einleitung LINK), wird das Programm mit der entsprechenden Fehlermeldung abgebrochen. Ansonsten werden die Files (in unserem Beispiel aus df0:c) an die Parameterliste angehängt und die Batchdatei unter demselben Namen wieder zurückgeschrieben. (br)

Listings

```
;*********************************************************
;***                      LinkList                     ***
;***             written by Willi Baecker              ***
;***              (c) 1990 by DMV-Verlag               ***
;***            Sprache: Assembler (Seka)              ***
;*********************************************************

Open            =       -30
Close           =       -30 -6
Read            =       -30 -12
Write           =       -30 -18
Output          =       -30 -30
Seek            =       -30 -36
Lock            =       -30 -54
UnLock          =       -30 -60
CurrentDir      =       -30 -96
Examine         =       -30 -72
ExNext          =       -30 -78

OpenLibrary     =       -30 -522
CloseLibrary    =       -30 -384
AllocMem        =       -30 -168
FreeMem         =       -30 -180
FindTask        =       -30 -264

EXEC:           equ     4
CSI:            equ     $9b
Mode_old:       equ     1005
Mode_new:       equ     1006

start:
        subq    #1,d0
        beq     usage                   ;branch if no parameter
        andi.l  #$ff,d0                 ;clear register
        move.b  d0,paralength           ;store length
        cmp.w   #$3f0a,(a0)
        beq     usage
        subq    #1,d0                   ;for dbra
        lea     path(pc),a1             ;namebuffer
loop1:  move.b  (a0)+,(a1)+             ;copy cliParameter to buffer
        cmp.b   #10,(a0)                ;RETURN found?
        beq     next                    ;branch if next
        dbra    d0,loop1                ;next digit
next:   lea     Path(pc),a0             ;get buffer again
        moveq   #0,d0                   ;clear register
        move.b  paralength,d0           ;get parameterlength
loop2:  cmp.b   #" ",(a0)+              ;is there an option?
        beq.s   found                   ;Yes there is!
        dbra    d0,loop2                ;test next digit
        bra     nooption                ;no option
found:  cmp.b   #10,(a0)                ;correct option?
        beq     IllegalOption           ;no, go to hell!
        clr.b   -1(a0)                  ;clear 1st comma
        move.l  a0,OptionName           ;store adress of OptName
        lea     OptionFbuffer(pc),a1
ff1:    move.b  (a0)+,(a1)+
        tst.b   (a0)
        bne.s   ff1
        st      LinkFlag                ;set a Flag
nooption:
        bsr     Opendos                 ;OpenLib + CliHandle
        tst.b   LinkFlag                ;was Option there?
        beq     directory               ;branch if not

        move.l  DosBase(pc),a6          ;get Baseadress
        move.l  OptionName(pc),d1       ;get adress of Filename
        move.l  #Mode_old,d2            ;I want to READ
        jsr     Open(a6)                ;Open file
        beq     Nooptionfile            ;branch on error
        move.l  d0,OptionHandle         ;store handle
        move.l  DosBase(pc),a6
        move.l  OptionHandle(pc),d1
        moveq   #0,d2
        moveq   #1,d3
        jsr     Seek(a6)
```

Listing: LinkList

```
        move.l  DosBase(pc),a6
        move.l  OptionHandle(pc),d1
        moveq   #0,d2
        moveq   #0,d3
        jsr     Seek(a6)
        move.l  d0,OptionFilelength
        move.l  DosBase(pc),a6
        move.l  OptionHandle(pc),d1
        moveq   #0,d2
        moveq   #-1,d3
        jsr     Seek(a6)
        move.l  EXEC,a6
        move.l  OptionFilelength(pc),d0
        add.l   #30000,d0
        move.l  #$10002,d1
        jsr     AllocMem(a6)
        beq     MemoryError
        move.l  d0,memad0
        move.l  DosBase(pc),a6
        move.l  OptionHandle(pc),d1
        move.l  memad0,d2
        move.l  OptionFileLength,d3
        jsr     Read(a6)
        move.l  DosBase(pc),a6
        move.l  OptionHandle(pc),d1
        jsr     Close(a6)
        move.l  memad0(pc),a0
        moveq   #3,d0
test1:  cmp.b   #"a",(a0)
        bhi.s   test11
        add.b   #$20,(a0)+
test11: dbra    d0,test1
        move.l  memad0(pc),a0
        cmp.l   #"link",(a0)            ;Test File
        bne     IncorrectFile
        move.l  memad0(pc),d0
        add.l   OptionFilelength,d0
        move.l  d0,newbuffer
directory:
        move.l  DosBase(pc),a6          ;get LibBase
        move.l  #path,d1                ;get adress of Dirname
        moveq   #-2,d2                  ;mode: read
        jsr     Lock(a6)
        beq     error                   ;branch on error
        move.l  d0,locksave             ;store new Lock
        move.l  DosBase(pc),a6
        move.l  locksave(pc),d1
        move.l  #fileinfo,d2
        jsr     Examine(a6)
        beq     error                   ;ready
        lea     fileinfo+4(pc),a0
        cmp.l   #-3,(a0)                ;is it a File?
        beq     Fileerror               ;Yes, branch!

        move.l  #Return,d0
        bsr     DosPrint
        move.l  #slash,d0
        bsr     DosPrint
        move.l  #path,d0
        bsr     DosPrint
        move.l  #Return,d0
        bsr     DosPrint
        tst.b   LinkFlag
        beq     loop
        move.l  newbuffer(pc),a3        ;never trash a3
        move.b  #10,(a3)+
        move.b  #"\",(a3)+
        lea     path(pc),a0
copy:   move.b  (a0)+,(a3)+
        cmp.b   #0,(a0)
        bne.s   copy
        move.b  #10,(a3)+
loop:
        move.l  mytask(pc),a4
        btst    #12,28(a4)              ;Control C
        beq.s   control
        move.l  #ExitText,d0
        bsr     DosPrint
        bra     Error
control:lea     ziffbuffer(pc),a0
        clr.l   (a0)+
```

Listing: LinkList

```
                clr.l   (a0)+
                move.l  DosBase(pc),a6
                move.l  locksave(pc),d1
                move.l  #fileInfo,d2
                jsr     ExNext(a6)
                tst.l   d0
                beq     nofilemore
                lea     fileinfo(pc),a0
                move.l  120(a0),d3
                cmp.b   #-3,d3
                bne     noz
                move.l  #fileinfo+8,d0
                bsr     dosprint
                tst.b   LinkFlag
                beq     copy1
                move.l  #fileinfo+8,a0
        copy2:  move.b  (a0)+,(a3)+
                cmp.b   #0,(a0)
                bne.s   copy2
        copy1:  move.l  d0,a0
                move.l  d0,d1
        nameloop:
                addq    #1,d0
                tst.b   (a0)+
                bne.s   nameloop
                sub.l   d1,d0
                subq    #1,d0
                moveq   #0,d2
                moveq   #33,d1
                sub.l   d0,d1
                divu    #8,d1
                move.w  d1,d2                   ;d2 = tabs
                lea     textbuffer(pc),a0
                swap    d1
                tst.w   d1
                beq.s   nospace
                andi.l  #$ffff,d1
                subq    #1,d1
        sploop: lea     space(pc),a1
                move.b  (a1),(a0)+
                dbra    d1,spLoop
        nospace:
                tst.w   d2
                beq.s   notabs
                subq    #1,d2
        tabloop:lea     tab(pc),a1
                move.b  (a1),(a0)+
                dbra    d2,tabloop
        notabs:
                move.l  #textbuffer,d0
                bsr     DosPrint
        test:   tst.b   LinkFlag
                beq.s   copy4
                lea     Textbuffer(pc),a0
        copy3:  move.b  (a0)+,(a3)+
                cmp.b   #0,(a0)
                bne.s   copy3
        copy4:  lea     fileinfo(pc),a0
                move.l  124(a0),d4
                tst.l   d4
                beq     noziffer
                lea     ziffbuffer(pc),a5
                bsr     bindez
                move.l  #ziffbuffer,d0
                bsr     dosprint
                tst.b   LinkFlag
                beq     noziffer
                lea     ziffbuffer(pc),a0
        copy6:  move.b  (a0)+,(a3)+
                cmp.b   #0,(a0)
                bne.s   copy6
                move.b  #10,(a3)+
                move.l  a3,Endefile
        noziffer:
                move.l  #return,d0
                bsr     dosprint
        noz:
                lea     Textbuffer(pc),a0
                moveq   #79,d0
        clrbuff:clr.b   (a0)+
                dbra    d0,clrbuff
                bra     loop
        error:
                tst.l   memad0
                beq.s   noFreeMem
                move.l  EXEC,a6
                move.l  memad0(pc),a1
                move.l  OptionFilelength,d0
                add.l   #30000,d0
                jsr     FreeMem(a6)
        noFreemem:
                move.l  #return,d0
                bsr     dosprint
                bsr     CloseDos
        noparas:
                moveq   #0,d0
                sub.l   a0,a0
                rts
        ;------------------------------------------------------
        nofilemore:
                tst.b   LinkFlag
                beq.s   Error
                move.l  DosBase(pc),a6
                move.l  #OptionFBuffer,d1
                move.l  #mode_new,d2
                jsr     Open(a6)
                move.l  d0,writehandle
                move.l  d0,d1
                move.l  endefile,d3
                move.l  memad0,d2
                sub.l   d2,d3
```

Listing: LinkList

```
         jsr     Write(a6)
         bmi     writeError
back:    move.l  writehandle,d1
         jsr     Close(a6)
         bra     error
;-------------------------------------------------------------
writeerror:
         move.l  #writeErrText,d0
         bsr     DosPrint
         bra     back
;-------------------------------------------------------------
usage:
         bsr     OpenDos
         move.l  #UsageText,d0
         bsr     DosPrint
         bsr     CloseDos
         moveq   #0,d0
         sub.l   a0,a0
         rts

;-------------------------------------------------------------
IllegalOption:
         bsr     OpenDos
         move.l  #IllegalText,d0
         bsr     DosPrint
         bsr     CloseDos
         sub.l   a0,a0
         moveq   #0,d0
         rts
;-------------------------------------------------------------
IncorrectFile:
         move.l  #IncorrectText,d0
         bsr     DosPrint
         bra     Error
;-------------------------------------------------------------
FileError:
         move.l  #FileErrorText,d0
         bsr     DosPrint
         bra     Error
;-------------------------------------------------------------
MemoryError:
         move.l  #MemErrorText,d0
         bsr     DosPrint
         move.l  DosBase(pc),a6
         move.l  OptionHandle(pc),d1
         jsr     Close(a6)
         bsr     CloseDos
         moveq   #0,d0
         sub.l   a0,a0
         rts

;-------------------------------------------------------------
;-------------------------------------------------------------
Nooptionfile:
         move.l  #Optionfileerror,d0
         bsr     DosPrint
         move.l  OptionName(pc),d0
         bsr     DosPrint
         move.l  #Return,d0
         bsr     DosPrint
         bsr     CloseDos
         sub.l   a0,a0
         moveq   #0,d0
         rts
;-------------------------------------------------------------
OpenDos:
         move.l  4,a6
         sub.l   a1,a1
         jsr     FindTask(a6)
         move.l  d0,mytask
         move.l  4,a6
         lea     DosName(pc),a1
         moveq   #0,d0
         jsr     OpenLibrary(a6)
         move.l  d0,DosBase
         move.l  d0,a6
         jsr     Output(a6)
         move.l  d0,CliHandle
         rts
;-------------------------------------------------------------
CloseDos:
         move.l  4,a6
         move.l  DosBase(pc),a1
         jsr     CloseLibrary(a6)
         rts
;-------------------------------------------------------------
DosPrint:
         movem.l d0-d3/a0/a6,-(sp)   ;alle Register auf STACK
         move.l  d0,d2               ;start retten
         move.l  d2,d3
         move.l  d2,a0               ;String-adresse uebergeben
DosPrint0:
         tst.b   (a0)+               ;Byte testen & Adresse + 1
         bne.s   DosPrint0           ;Sprung wenn nicht null
         sub.l   a0,d3               ;
         neg.l   d3                  ;positiv machen
         subq    #1,d3
         move.l  CliHandle(pc),d1
         move.l  DosBase,a6          ;Basisadresse holen
         jsr     Write(a6)           ;(dos.library) = -48(a6)
         movem.l (sp)+,d0-d3/a0/a6   ;Register vom STACK holen
         rts                         ;zurueck
;-------------------------------------------------------------
;-------------------------------------------------------------
;Binaerzahl in DezimalString wandeln
;a5= Zeiger auf StringBuffer
;Zahl steht in d4
BinDez:  movem.l d0-d7/a0-a6,-(sp)
         tst.l   d4                  ;Zahl in d4
         beq.s   BinDez6             ;branch if null
         bmi.s   BinDez1             ;branch if minus

         neg.l   d4                  ;neg it
         bra.s   BinDez2
BinDez1:move.b   #'-',(a5)+
```

Listing: LinkList

```
BinDez2:lea      Decathlon,a0
         clr.w   d1
BinDez3:move.l   (a0)+,d2
         beq.s   BinDez6
         moveq   #-1,d0
BinDez4:add.l    d2,d4
         dbgt    d0,BinDez4
         sub.l   d2,d4
         addq.w  #1,d0
         bne.s   BinDez5
         tst.w   d1
         beq.s   BinDez3
BinDez5:moveq    #-1,d1
         neg.b   d0
         addi.b  #$30,d0
         move.b  d0,(a5)+
         bra.s   BinDez3
BinDez6:neg.b    d4
         addi.b  #$30,d4
         move.b  d4,(a5)+
         movem.l (sp)+,d0-d7/a0-a6
         rts
;-------------------------------------------------------------
Decathlon:
         dc.l    1000000000
         dc.l    100000000
         dc.l    10000000
         dc.l    1000000
         dc.l    100000
         dc.l    10000
         dc.l    1000
         dc.l    100
         dc.l    10
         dc.l    0
;-------------------------------------------------------------
align    4
fileinfo:          blk.l   260,0
path:              blk.b   200,0
ziffbuffer:        dc.l    0,0,0,0
textbuffer:        blk.b   80,0
OptionFBuffer:     blk.b   80,0
;-------------------------------------------------------------
locksave:          dc.l    0
DosBase:           dc.l    0
memad0:            dc.l    0
writehandle:       dc.l    0
clihandle:         dc.l    0
OptionHandle:      dc.l    0
OptionFilelength:dc.l      0
OptionName:        dc.l    0            ;adress of OptFilename
newbuffer:         dc.l    0
EndeFile:          dc.l    0
mytask:            dc.l    0
paralength:        dc.b    0,0          ;len of parameters
LinkFlag:          dc.b    0,0
;-------------------------------------------------------------
DosName:           dc.b    "dos.library",0
;-------------------------------------------------------------
Return:            dc.b    10,0
;-------------------------------------------------------------
slash:             dc.b    "\",0
;-------------------------------------------------------------

space:             dc.b    32,0
;-------------------------------------------------------------
tab:               dc.b    9,0
;-------------------------------------------------------------
Illegaltext:       dc.b    10
                   dc.b    "ERROR !",10
                   dc.b    "Illegal option recognized !",10
                   dc.b    0
;-------------------------------------------------------------
OptionFileerror:dc.b       10
                   dc.b    "ERROR !",10
                   dc.b    "Unable to open Parameterfile > "
                   dc.b    0
;-------------------------------------------------------------
MemErrorText:      dc.b    10,10
                   dc.b    "ERROR !",10
                   dc.b    "Cannot allocate needed memory size.",10
                   dc.b    "Reset your AMIGA and try again!",10
                   dc.b    0
;-------------------------------------------------------------
WriteErrText:      dc.b    10,10
                   dc.b    "ERROR !",10
                   dc.b    "Cannot write parameterfile!",10
                   dc.b    0
;-------------------------------------------------------------
UsageText:
                   dc.b    "****************************************",10
                   dc.b    "Usage: LinkList DFx:[ Parameterfilename]",10
                   dc.b    "****************************************",10
                   dc.b    "A FACTOR 5 production 1989 by W.Baecker!",10
                   dc.b    "****************************************"
                   dc.b    10,0
;-------------------------------------------------------------
ExitText:          dc.b    "^C",10,0
;-------------------------------------------------------------
ExitText1:         dc.b    "**BREAK",10,0
;-------------------------------------------------------------
IncorrectText:     dc.b    10,"ERROR:",10
                   dc.b    "This is not a valid Parameter File.",10
                   dc.b    "Be sure that your ParameterFile starts"
                   dc.b    " with  LINK  !",10,0
;-------------------------------------------------------------
FileErrorText:     dc.b    10,"ERROR:",10
                   dc.b    "This is not a directory !",0
;-------------------------------------------------------------
                   even
```

Listing: LinkList

Garry Glendown

Dem CLI Beine machen

Im Blickpunkt: die ARP-Library

Bei der Programmierung des Amiga gibt es kleine, mittlere und große Schwierigkeiten, die man je nach persönlichem Kenntnisstand besser oder schlechter bewältigen kann. Heraus kommt dann die x-te Speicherverwaltungsroutine oder die 200ste Pattern-Match-Routine. Es ist schon schlimm, daß der Amiga das nicht standardmäßig zur Verfügung stellt…

Doch ganz so schlimm ist es nun ja auch nicht, denn gemessen an dem Grad der Verbreitung gibt es einen 'Quasi-Standard', der mehr und mehr von den Programmierern genutzt wird: die ARP-Library.

Doch was ist dieses ARP eigentlich? Die ARP-Distribution besteht neben der eigentlichen Library aus der kompletten Dokumentation zu den Routinen, einem Satz DOS-Befehle, die praktisch durchgängig besser als die von Commodore gelieferten sind (natürlich auch inklusive Dokumentation), sowie den Routinen zur Einbindung in verschiedene Programmiersprachen. Neben diesen Sprachen ist aber das Einbinden in andere Umgebungen eigentlich kein Problem, so daß auch eine Nutzung von ARP innerhalb von Amiga- oder GFA-BASIC möglich ist.

Von den Befehlen…

Die ARP-DOS-Befehle zeichnen sich dadurch aus, daß sie zum einen alle kürzer sind als ihre Amiga-DOS-Äquivalente, zum anderen eine bessere Leistung an den Tag legen. Angefangen mit dem "*" als Wildcard, den die ARP-Befehle zusätzlich besitzen, über die geringere Anfälligkeit bei Sonderfällen (Beispiel: Geht man mit dem Amiga-DOS-"Search" auf ein Binärfile los, so wird man recht viele "Warning: line <x> to long" bekommen, in denen dann natürlich auch nicht gesucht wird. ARP-Search geht da anstandslos durch…) bis hin zu den wesentlich größeren Möglichkeiten (Beispiel: ARP-"Sort" ist nicht nur schneller, sondern man hat noch die Möglichkeit, die Sortierbreite und eine Sortierung mit Unterscheidung von Groß- und Kleinschreibung zu bestimmen).

Leider enthalten die Befehle der aktuellen Version noch einige Fehler, die jedoch in der Praxis in Anbetracht der besseren Leistung in Kauf genommen werden können.

Doch wie haben es die Programmierer der Befehle überhaupt geschafft, die Befehle kürzer zu machen? Statt Routinen wie die Abfrage der Übergabeparameter in jedem Programm abzulegen, wird einfach eine Routine in der ARP-Library verwendet. Werden dann später Erweiterungen oder Fehlerkorrekturen vorgenommen, zum Beispiel beim Pattern-Matching oder ähnlichem, so sind automatisch alle Programme korrigiert, wenn die Library ausgetauscht wird. So laufen natürlich Programme, die mit ARP 1.1 programmiert wurden, auch mit der Version 1.3 einwandfrei. Funktionen wie der Filerequester zeigen nicht mehr den alten, sondern den neuen Requester.

…zur Library

Der Datenteil der Library-Struktur, also die eigentliche Librarybase, enthält, im Vergleich zu den Möglichkeiten der ARP-Library, recht wenig Daten. Interessant sind lediglich die Zeiger auf DOS-, Graphics- und Intuition-Library, die man bei erfolgreichem Öffnen der ARP-Library nur noch übernehmen muß.

Dagegen ist jedoch die Größe der Sprungtabelle wahrhaft riesig; sie übertrifft mit ihren 105 Routinen sogar die schon recht große Graphics-Library. In Tabelle 1 befindet sich eine Auflistung aller Routinen mit Offset sowie den Parametern (mit Registerangabe für Assemblerprogrammierer).

Um nun eigene Programme mit diesen Routinen 'anzureichern', wollen wir an dieser Stelle erst einmal systematisch nachsehen, welche Routinen besonders interessant sind. Dazu müssen natürlich Vorbereitungen getroffen werden. Entweder man benutzt die von der "arp.lib" bereitgestellten Routinen (in diesem Fall wird die Library etc. automatisch geöffnet), oder man öffnet die ARP-Library selber. In diesem Fall muß man mit "a.lib" linken.

Doch nun zu den 'wirklichen' Library-Routinen. Als erstes fallen da natürlich die Routine "Printf", "FPrintf" und "SPrintf" ins Auge. Diese sorgen bei normaler Anwendung der Routinen aus der "c.lib" dafür, daß jedes beliebige Programm automatisch um einiges wächst. Zur Verwendung braucht man, dank Argument-Kompatibilität, nur die Namen in den Aufrufen zu ändern oder aber Zeilen wie:

```
#define printf Printf
#define sprintf SPrintf
#define fprintf FPrintf
```

"Im selben Aufwasch" entledigen wir uns noch der Puts-Funktion, die wir durch "Puts" ersetzen. Nicht zu vergessen die Routinen zum String-Vergleich, "Strcmp" und "Strncmp". Also noch drei Defines:

```
#define puts Puts
#define strcmp Strcmp
#define strncmp Strncmp
```

Zum Abschluß wollen wir noch einen Blick auf ein recht heikles Kapitel werfen: die Pattern-Match-Routinen, deren vielzählige Varianten man mit dem Satz "oft kopiert – nur selten erreicht" charakterisieren kann. Sie kennen das Problem: Wie oft war man schon der Meinung, die wirklich optimale Routine gefunden zu haben, und dann trat irgendwann ganz nebenbei ein Fall auf, bei der sie mit Pauken und Trompeten "durch die Prüfung gerasselt" ist… Ganz zu schweigen von einigen Funktionen, bei der nur bestimmte Zeichen zugelassen sind, und vieles mehr.

Alles das wird von den ARP-Routinen doch recht perfekt beherrscht. Und da das Ganze noch recht einfach einzusetzen ist, wollen wir uns hier ein kurzes Code-Fragment anschauen, in dem diese Routinen angewandt werden.

Gegeben sei ein String "a", dessen Vorkommen wir in verschiedenen eingelesenen Zeilen testen wollen. Zu diesem Zweck muß er noch vorbereitet werden. Dies geschieht folgendermaßen: Wir kopieren ihn in einen anderen String-Puffer hinein, wobei vorne und hinten noch ein '*' angefügt wird. Um auch noch unabhängig von Groß- und Kleinschreibung zu sein, wird er vorher noch mittels "StrToUpper" in Großschrift umgewandelt:

```
VOID StrToUpper(txt)
char *txt;
{
   while (*txt) {
      *txt=Toupper(*txt);
      txt++;
   }
}
:
:
int Match(a)
char *a;
{
char b[256],c[256];
:
   strcpy(b,"*");
   strcat(b,a);
   strcat(b,"*");
   StrToUpper(b);
```



Nun steht also in "b" der gesuchte Text in Großschrift. Damit wir ihn mit der Pattern-Match-Routine verwenden können, muß er noch bearbeitet werden, was mit "PreParse" gemacht wird. Dabei werden die verschiedenen Darstellungsarten auf ein internes Format gebracht:

```
PreParse(a,c);
```

Der String in "c" kann nun mittels der Routine "PatternMatch" zum Vergleichen verwendet werden. Bei einem Array in "nums" von "g"-Strings kann nun beispielsweise mit folgender Schleife durchsucht werden:

```
for (t=0;t<g;t++) {
  strcpy(b,nums[t]);
  StrToUpper(b);
  if (PatternMatch(c,b))
   Printf("Gefunden: %s\n",
   nums[t]);
}
```

Damit beenden wir den ersten Teil unseres kleinen ARP-Exkurses. In der nächsten Ausgabe der AMIGA DOS gehen wir auf die Programmierung des Filerequesters sowie einiger anderer Routinen ein.

(vb)

Alphabetische Liste der ARP-Routinen

AddDADevs	$-204	-516 dalist/A0, Select/D0
AddDANode	$-1FE	-510 Data/A0, *DALst/A1, Len/D0, ID/D1
AddResidentPrg	$-246	-582 Segment/D1, Name/A0
ArpAlloc	$-180	-384 Size/D0
ArpAllocFreq	$-294	-660 -
ArpAllocMem	$-186	-390 Size/D0, Reqs/D1
ArpDupLock	$-192	-402 Lock/D1
ArpExit	$-17A	-378 ReturnCode/D0, Fault/D2
ArpLock	$-198	-408 Name/D1, Accessmode/D2
ArpOpen	$-18C	-396 Name/D1, Accessmode/D2
ArpOpenLibrary	$-28E	-654 Lib/A1, Version/D0
Assign	$-150	-336 Name/A0, Physical/A1
ASyncRun	$-222	-546 Command/A0, Args/A1, PCB/A2
Atol	$-102	-258 String/A0
BaseName	$-276	-630 PathName/A0
BtoCStr	$-162	-354 CStr/A0, BStr/D0, MaxLen/D1
CheckAbort	$-10E	-270 (*Func)()/A1
CheckBreak	$-114	-276 Mask/D1, (*Func)()/A1
CheckSumPrg	$-26A	-618 Node/D1
Close	$-024	-36 /* = DOS-Funktion */
CloseWindowSafely	$-12C	-300 Window/A0, MoreWindows/A1
CompareLock	$-1C8	-456 Lock1/D0, Lock2/D1
CreateDir	$-078	-120 /* = DOS-Funktion */
CreatePort	$-132	-306 Name/A0, Priority/D0(0:8)
CreateProc	$-08A	-138 /* = DOS-Funktion */
CreateTaskResList	$-1D4	-468 -
CtoBStr	$-168	-360 CStr/A0, BStr/D0, MaxLen/D1
CurrentDir	$-07E	-126 /* = DOS-Funktion */
DateStamp	$-0C0	-192 /* = DOS-Funktion */
Delay	$-0C6	-198 /* = DOS-Funktion */
DeleteFile	$-048	-72 /* = DOS-Funktion */
DeletePort	$-138	-312 Port/A1
DeviceProc	$-0AE	-174 /* = DOS-Funktion */
DosAllocMem	$-156	-342 Size/D0
DosFreeMem	$-15C	-348 MemBlock/A1
DupLock	$-060	-96 /* = DOS-Funktion */
EscapeString	$-108	-264 String/A0
Examine	$-066	-102 /* = DOS-Funktion */
Execute	$-0DE	-222 /* = DOS-Funktion */
Exit	$-090	-144 /* = DOS-Funktion */
ExNext	$-06C	-108 /* = DOS-Funktion */
FileRequest	$-126	-294 FileRequester/A0
FindCLI	$-1A4	-420 TaskNum/D0
FindFirst	$-1B6	-438 Pattern/D0, AnchorPath/A0
FindNext	$-1BC	-444 AnchorPath/A0
FindTaskResList	$-1CE	-462 -
FPrintf	$-0EA	-234 File/D0, String/A0, *args/A1
FreeAccess	$-1F2	-498 Tracker/A1
FreeAnchorChain	$-1C2	-450 AnchorPath/A0
FreeDAList	$-1F8	-504 daNode/A1
FreeResList	$-1DA	-474 -
FreeTaskResList	$-174	-372 -
FreeTrackedItem	$-1E0	-480 Tracker/A1
GADS	$-0FC	-252 line/A0, len/D0, help/A1, args/a2, tplate/A3
GetAccess	$-1EC	-492 Tracker/A1
GetDevInfo	$-16E	-366 DevInfoNode/A2
Getenv	$-11A	-282 String/A0, Buffer/A1, Size/D0
GetKeywordIndex	$-288	-648 KWrd/A0, template/A1
GetTracker	$-1E6	-486 -
Info	$-072	-114 /* = DOS-Funktion */
InitStdPacket	$-144	-324 Action/D0, Args/A0, Packet/A1, ReplyPort/A2
Input	$-036	-54 /* = DOS-Funktion */
IoErr	$-084	-132 /* = DOS-Funktion */
IsInteractive	$-0D8	-216 /* = DOS-Funktion */
LDiv	$-25E	-606 Dividend/D0, Divisor/D1
LMod	$-264	-612 Dividend/D0, Divisor/D1
LMult	$-258	-600 Num1/D0, Num2/D1
LoadPrg	$-228	-552 Name/D1
LoadSeg	$-096	-150 /* = DOS-Funktion */
Lock	$-054	-84 /* = DOS-Funktion */
ObtainResidentPrg	$-240	-576 Name/A0
Open	$-01E	-30 /* = DOS-Funktion */
Output	$-03C	-60 /* = DOS-Funktion */
ParentDir	$-0D2	-210 /* = DOS-Funktion */
PathName	$-14A	-330 Lock/D0, Dest/A0, NumberNames/D1
PatternMatch	$-1B0	-432 Pattern/A0, String/A1
PreParse	$-22E	-558 Source/A0, Dest/A1
Printf	$-0E4	-228 String/A0, *args/A1
Puts	$-0F0	-240 String/A1
QSort	$-1AA	-426 Ptr/A0, RegSize/D0, ByteSize/D1, UserFunc/A1
Read	$-02A	-42 /* = DOS-Funktion */
ReadLine	$-0F6	-246 Buffer/A0
ReleaseResidentPrg	$-27C	-636 Segment/D1
RemResidentPrg	$-24C	-588 Name/A0
Rename	$-04E	-78 /* = DOS-Funktion */
RListAlloc	$-19E	-414 ResList/A0, Size/D0
Seek	$-042	-66 /* = DOS-Funktion */
SendPacket	$-13E	-318 Action/D0, Args/A0, Handler/A1
SetComment	$-0B4	-180 /* = DOS-Funktion */
Setenv	$-120	-288 String/A0, Buffer/A1
SetProtection	$-0BA	-186 /* = DOS-Funktion */
SpawnShell	$-222	-546
SPrintf	$-282	-642 buffer/D0, String/A0, *args/A1
StamptoStr	$-234	-564 DateTime/A0
Strcmp	$-20A	-522 S1/A0, S2/A1
Strncmp	$-210	-528 S1/A0, S2/A1, Len/D0
StrtoStamp	$-23A	-570 DateTime/A0
SyncRun	$-21C	-540 FileName/A0, Args/A1, Input/D0, Output/D1
TackOn	$-270	-624 Path/A0, File/A1
Toupper	$-216	-534 Char/D0
UnLoadPrg	$-252	-594 Segment/D1
UnLoadSeg	$-09C	-156 /* = DOS-Funktion */
UnLock	$-05A	-90 /* = DOS-Funktion */
WaitForChar	$-0CC	-204 /* = DOS-Funktion */
Write	$-030	-48 /* = DOS-Funktion */

Lister ARP-Befehle, nach Offset sortiert

Open	$-01E	-30 /* = DOS-Funktion */
Close	$-024	-36 /* = DOS-Funktion */
Read	$-02A	-42 /* = DOS-Funktion */
Write	$-030	-48 /* = DOS-Funktion */
Input	$-036	-54 /* = DOS-Funktion */
Output	$-03C	-60 /* = DOS-Funktion */
Seek	$-042	-66 /* = DOS-Funktion */
DeleteFile	$-048	-72 /* = DOS-Funktion */
Rename	$-04E	-78 /* = DOS-Funktion */
Lock	$-054	-84 /* = DOS-Funktion */
UnLock	$-05A	-90 /* = DOS-Funktion */
DupLock	$-060	-96 /* = DOS-Funktion */
Examine	$-066	-102 /* = DOS-Funktion */
ExNext	$-06C	-108 /* = DOS-Funktion */
Info	$-072	-114 /* = DOS-Funktion */
CreateDir	$-078	-120 /* = DOS-Funktion */
CurrentDir	$-07E	-126 /* = DOS-Funktion */
IoErr	$-084	-132 /* = DOS-Funktion */
CreateProc	$-08A	-138 /* = DOS-Funktion */
Exit	$-090	-144 /* = DOS-Funktion */
LoadSeg	$-096	-150 /* = DOS-Funktion */
UnLoadSeg	$-09C	-156 /* = DOS-Funktion */
DeviceProc	$-0AE	-174 /* = DOS-Funktion */
SetComment	$-0B4	-180 /* = DOS-Funktion */
SetProtection	$-0BA	-186 /* = DOS-Funktion */
DateStamp	$-0C0	-192 /* = DOS-Funktion */
Delay	$-0C6	-198 /* = DOS-Funktion */
WaitForChar	$-0CC	-204 /* = DOS-Funktion */
ParentDir	$-0D2	-210 /* = DOS-Funktion */
IsInteractive	$-0D8	-216 /* = DOS-Funktion */
Execute	$-0DE	-222 /* = DOS-Funktion */
Printf	$-0E4	-228 String/A0, *args/A1
FPrintf	$-0EA	-234 File/D0, String/A0, *args/A1
Puts	$-0F0	-240 String/A1
ReadLine	$-0F6	-246 Buffer/A0
GADS	$-0FC	-252 line/A0, len/D0, help/A1, args/a2, tplate/A3
Atol	$-102	-258 String/A0
EscapeString	$-108	-264 String/A0
CheckAbort	$-10E	-270 (*Func)()/A1
CheckBreak	$-114	-276 Mask/D1, (*Func)()/A1
Getenv	$-11A	-282 String/A0, Buffer/A1, Size/D0
Setenv	$-120	-288 String/A0, Buffer/A1
FileRequest	$-126	-294 FileRequester/A0
CloseWindowSafely	$-12C	-300 Window/A0, MoreWindows/A1
CreatePort	$-132	-306 Name/A0, Priority/D0(0:8)
DeletePort	$-138	-312 Port/A1
SendPacket	$-13E	-318 Action/D0, Args/A0, Handler/A1
InitStdPacket	$-144	-324 Action/D0, Args/A0, Packet/A1, ReplyPort/A2
PathName	$-14A	-330 Lock/D0, Dest/A0, NumberNames/D1
Assign	$-150	-336 Name/A0, Physical/A1
DosAllocMem	$-156	-342 Size/D0
DosFreeMem	$-15C	-348 MemBlock/A1
BtoCStr	$-162	-354 CStr/A0, BStr/D0, MaxLen/D1
CtoBStr	$-168	-360 CStr/A0, BStr/D0, MaxLen/D1
GetDevInfo	$-16E	-366 DevInfoNode/A2
FreeTaskResList	$-174	-372 -
ArpExit	$-17A	-378 ReturnCode/D0, Fault/D2
ArpAlloc	$-180	-384 Size/D0
ArpAllocMem	$-186	-390 Size/D0, Reqs/D1
ArpOpen	$-18C	-396 Name/D1, Accessmode/D2
ArpDupLock	$-192	-402 Lock/D1
ArpLock	$-198	-408 Name/D1, Accessmode/D2
RListAlloc	$-19E	-414 ResList/A0, Size/D0
FindCLI	$-1A4	-420 TaskNum/D0
QSort	$-1AA	-426 Ptr/A0, RegSize/D0, ByteSize/D1, UserFunc/A1
PatternMatch	$-1B0	-432 Pattern/A0, String/A1
FindFirst	$-1B6	-438 Pattern/D0, AnchorPath/A0
FindNext	$-1BC	-444 AnchorPath/A0
FreeAnchorChain	$-1C2	-450 AnchorPath/A0
CompareLock	$-1C8	-456 Lock1/D0, Lock2/D1
FindTaskResList	$-1CE	-462 -
CreateTaskResList	$-1D4	-468 -
FreeResList	$-1DA	-474 -
FreeTrackedItem	$-1E0	-480 Tracker/A1
GetTracker	$-1E6	-486 -
GetAccess	$-1EC	-492 Tracker/A1
FreeAccess	$-1F2	-498 Tracker/A1
FreeDAList	$-1F8	-504 daNode/A1
AddDANode	$-1FE	-510 Data/A0, *DALst/A1, Len/D0, ID/D1
AddDADevs	$-204	-516 dalist/A0, Select/D0
Strcmp	$-20A	-522 S1/A0, S2/A1
Strncmp	$-210	-528 S1/A0, S2/A1, Len/D0
Toupper	$-216	-534 Char/D0
SyncRun	$-21C	-540 FileName/A0, Args/A1, Input/D0, Output/D1
SpawnShell	$-222	-546
ASyncRun	$-222	-546 Command/A0, Args/A1, PCB/A2
LoadPrg	$-228	-552 Name/D1
PreParse	$-22E	-558 Source/A0, Dest/A1
StamptoStr	$-234	-564 DateTime/A0
StrtoStamp	$-23A	-570 DateTime/A0
ObtainResidentPrg	$-240	-576 Name/A0
AddResidentPrg	$-246	-582 Segment/D1, Name/A0
RemResidentPrg	$-24C	-588 Name/A0
UnLoadPrg	$-252	-594 Segment/D1
LMult	$-258	-600 Num1/D0, Num2/D1
LDiv	$-25E	-606 Dividend/D0, Divisor/D1
LMod	$-264	-612 Dividend/D0, Divisor/D1
CheckSumPrg	$-26A	-618 Node/D1
TackOn	$-270	-624 Path/A0, File/A1
BaseName	$-276	-630 PathName/A0
ReleaseResidentPrg	$-27C	-636 Segment/D1
SPrintf	$-282	-642 buffer/D0, String/A0, *args/A1
GetKeywordIndex	$-288	-648 KWrd/A0, template/A1
ArpOpenLibrary	$-28E	-654 Lib/A1, Version/D0
ArpAllocFreq	$-294	-660 -

Stefan Hochmuth

Wie sieht's denn da aus?

FileMonitor – Einblick in jedes Programm

Fast jeder zeigt heute Einsicht: Die Grenzen gehen auf, die Abrüstung findet statt, der Umweltschutz hält endlich Einzug – nur manche Amiga-Programme weigern sich immer noch hartnäckig, Einsicht in sich selbst zu geben. Dabei gibt es genug Anlässe, die Bytes näher zu besehen und, falls notwendig, zu ändern, so zum Beispiel bei fehlerhaften Disketten, die sich im DOS mit Read- oder Write-Error bemerkbar machen, oder bei Programmen, die mal eben auf die schnelle korrigiert werden müssen, ohne den Source-Code zum wiederholten Male anzeigen zu lassen. Wie man solche Momente gut übersteht? Ganz einfach – mit FileMonitor.

File-Monitore haben ein großes Anwendungsfeld. Man kann sie zum Eindeutschen, zum Patchen oder einfach mal zum Anschauen von Objektdateien verwenden. Auf dem Amiga sind gute File-Monitore jedoch Mangelware. Die meisten belegen über 30 kByte auf Diskette, da sie in C geschrieben sind. Die Geschwindigkeit läßt demnach natürlich auch zu wünschen übrig.

FileMonitor V2.0 – schnell und komfortabel

Der FileMonitor V2.0, der komplett in Assembler programmiert wurde, behebt diese Mängel und bietet zudem eine komfortable Benutzeroberfläche. Diese wird von einer Vielfalt an Funktionen begleitet. Nach dem Eintippen können Sie das Programm durch die Eingabe seines Dateinamens im CLI starten. Optional können Sie die zu ladende Datei angeben, also zum Beispiel "FileMonitor C:NewCLI". Ein Start von der Workbench ist ebenfalls möglich, wenn Sie vorher eine passende '.info'-Datei erstellt haben.

Haben Sie keinen Parameter angegeben, so erscheint ein Requester, der Sie auffordert, die zu editierende Datei anzugeben. Um hierbei die größtmögliche Benutzerfreundlichkeit zu erreichen, ohne daß das Programm zu lang wird, wurde zur Dateiauswahl die Dateiauswahlbox aus der 'arp.library' verwendet. ARP steht für "Amiga-DOS Replacement Project" und wurde von einer eifrigen Programmierergruppe geschaffen, um die CLI-Befehle durch kürzere Assemblerprogramme zu ersetzen. Da ARP Public Domain ist, besitzt es eine hohe Verbreitung und ist leicht zu besorgen. Es lohnt sich wirklich (LIST-Befehl mit 2500 Bytes? – Im ARP schon realisiert!). Aber auch wenn Sie die 'arp.library' nicht in Ihrem LIBS:-Verzeichnis haben, können Sie FileMonitor verwenden. Dann können Sie den Dateinamen über einen einfacheren Requester eingeben. Ein eingegebener Leerstring bedeutet Abbruch (falls Sie die Maus nicht zur Hand haben, um "Abbruch" anzuklicken). Wenn alles glatt gegangen ist, wird jetzt die Datei geladen und angezeigt.

Achtung! Machen Sie sich auf jeden Fall vorher eine Sicherheitskopie von der zu editierenden Datei!

Bewegung, Bewegung!

Als erstes ist es natürlich wichtig zu wissen, wie man den Cursor bewegt. Um den Cursor auf eine Stelle zu bewegen, die bereits im Display sichtbar ist, genügt ein einfacher Mausklick. Übrigens gibt es natürlich nicht nur einen Cursor, sondern deren zwei, die immer gleichzeitig bewegt werden (einer im Hex-Zahlenfeld, der andere im zugehörigen ASCII-Feld). Sehen Sie oben und unten im File-Monitor-Fenster die beiden orangeroten (Farbe 3), beschrifteten Leisten? Auf ihnen sind insgesamt sechs Funktionen dem Bewegen innerhalb der Datei gewidmet.

Angewählt werden diese durch einfaches Anklicken des zugehörigen Textes. Die Funktion wird dann so lange ausgeführt, wie man die linke Maustaste gedrückt hält. Damit kann man die Datei bequem mausgesteuert durchblättern (Seite vorwärts/rückwärts) oder durchscrollen (Zeile vorwärts/ rückwärts) bzw. den Cursor auf den Dateianfang oder das Dateiende setzen. All diese Funktionen sind auch über Tastatur erreichbar. Die Cursortasten sind wie in vielen Editoren belegt:

Cursortaste links/rechts/oben/unten
Cursor in entsprechende Richtung

SHIFT + oben
Eine Bildschirmseite rückwärts

SHIFT + unten
Eine Bildschirmseite vorwärts

ALT + oben/unten
Dateianfang/Dateiende

Auch über die Menüsteuerung im Positionsmenü läßt sich einiges erreichen:

Dateianfang
Rechte Amiga-Taste + T

Dateiende
Rechte Amiga-Taste + B

Hexoffset eingeben
Rechte Amiga-Taste + G

Auf die letzte Eingabe hin erscheint ein Requester, in dem Sie die gewünschte Dateiposition als Hexadezimalzahl angeben können. Nach Eingabe von 'Return' springt der Cursor an die gewünschte Stelle. Haben Sie eine 'Nicht-Hex-Zahl' angegeben, so erhalten Sie eine Fehlermeldung. Wenn die Hex-Zahl größer als die Dateilänge (wird übrigens links unten im FileMonitor neben dem Dateizeiger als Hexzahl angezeigt) sein sollte, erfolgt keine Reaktion.

Viele Wege führen zum Editieren

Der Editiermodus ist gewählt, wenn einer der beiden Cursor aus einem ausgefüllten Rechteck und nicht mehr nur aus einem Rahmen besteht. Die Hex- bzw. die ASCII-Hälfte befindet sich dann im Editiermodus. Um in diesen zu gelangen, gibt es einige Möglichkeiten:

A) Mit der Maus: Machen Sie einen Doppelklick an die Stelle im HEX- bzw. ASCII-Display, an der Sie mit dem Editieren beginnen möchten.

B) Über die Leiste: Klicken Sie 'Ed HEX' an, um die Hex-Hälfte zu editieren, bzw. 'Ed ASCII' für die Texthälfte.

C) Über Menü: Wählen Sie 'HEX-/ASCII-Hälfte editieren' oder 'Rechte Amiga-Taste + X / A' aus.

D) F10-Taste: Editiermodus an/aus

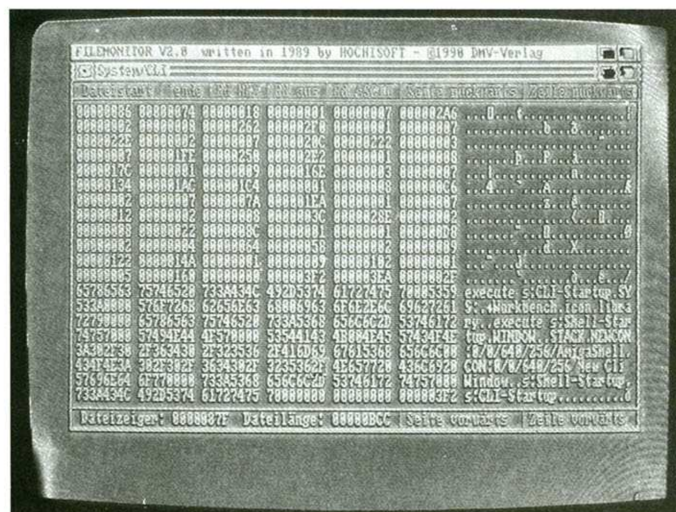


Bild 1. Mit dem FileMonitor auf Byte-Suche

E) TAB-Taste: Editiermodus anschalten und zwischen HEX-/ASCII-Editiermodus wechseln

Auch im Editiermodus besitzen Sie weiterhin alle Möglichkeiten, sich in der Datei zu bewegen. Das Editieren selbst erfolgt im ASCII-Modus durch einfache Eingabe, auf der Hex-Seite wird nur 0 bis 9 und A bis F akzeptiert. Das Ergebnis Ihrer Eingabe wird jedoch erst nach dem Tippen beider Hex-Nibble (= "Hex-Zahlen-Hälften") im Display erscheinen. Im Editiermodus kann man sehr leicht durch ungewollte Tastendrücke ebenso ungewollte Veränderungen vornehmen. Schalten Sie ihn deswegen nach dem Editieren vorsichtshalber sofort wieder aus, indem Sie entweder "Ed aus" mit der Maus in der Menüleiste anklicken oder "Editieren aus" im Editierenmenü anwählen (bzw. rechte Amiga-Taste + " – "). Drücken Sie bei der Short-cut-Wahl gewissenhaft die richtigen Tasten, um dabei nicht versehentlich zu editieren.

F10-Taste: Editiermodus an/aus

Wenn man beim Editieren einen Fehler macht, gibt es eine Möglichkeit, diesen wieder rückgängig zu machen. Und zwar mit der Seiten-Undo-Funktion aus dem Editierenmenü. Bei Aufruf wird alles Editierte, was in der Zwischenzeit nicht vom Bildschirm verschwunden ist, wieder in den vorigen Zustand gesetzt. Wenn Sie Routinen verwenden, die das ganze Display neu aufbauen, wird dieser interne Undo-Buffer folglich auch neu geladen. Meiden Sie deshalb folgende Funktionen, wenn Sie eventuell ein Undo machen wollen:

- Dateianfang/-ende
- Seite vor-/rückwärts
- Suchen
- Hexoffset eingeben

Bei normalen Cursorbewegungen (bzw. Zeile vor/zurück) wird der interne Buffer mitgescrollt, das heißt, wenn nach oben gescrollt wird, verschwindet die erste Zeile aus dem Buffer, alle anderen rücken eine Zeile nach und die neu erscheinende wird wieder in den Buffer geschrieben. Die Seiten-Undo-Funktion erweist sich manchmal als sehr nützlich.

Suchen und Finden

Selbstverständlich besitzt der FileMonitor auch diverse Funktionen zur String- und Hex-Zeichenfolgesuche. Mit den ersten beiden Menüpunkten im Suchen-Menü können Sie bestimmen, ob Sie eine Fallunterscheidung von Groß- und Kleinbuchstaben wünschen. Voreingestellt ist, daß Groß- und Kleinschreibung gleich behandelt wird. Bei "String suchen" erscheint wieder ein Requester, der Sie nach der zu suchenden Zeichenfolge fragt. Nach der Bestätigung mit Return-Taste wird nach dem String gesucht. Klicken Sie "Vorige!" an, um die vorhergehende Zeichenfolge zu suchen. Wird sie nicht gefunden, so bleibt der Cursor an der alten Stelle. Um ohne Neueingabe des Strings vor- oder rückwärts die gleiche Zeichenfolge suchen zu können, wählen Sie im Menü die entsprechenden Funktionen oder deren Short cuts an. Gleiches gilt für das Suchen von Hex-Folgen, allerdings müssen Sie bei deren Eingabe beachten, daß immer zwei Zeichen für ein ganzes Byte stehen. Eine Eingabe wie "3AB" wird daher zurückgewiesen.

Bleibt nur noch das Projektmenü zu erklären. Das Laden wurde bereits anfangs beschrieben. Die Bedienung der Speichern-Funktion ist entsprechend. Nochmals die Warnung: Arbeiten Sie nur mit Backups! Um einen Datenverlust zu vermeiden, sind im FileMonitor übrigens folgende Sicherheitsabfragen eingebaut:

A) Änderungen verlieren?
Wenn Sie eine Datei editiert haben und den FileMonitor verlassen oder eine neue Datei laden wollen, obwohl Sie vorher nicht gespeichert haben, erscheint diese Abfrage.

B) Alte Datei überschreiben?
Existiert die beim Speichern angegebene Datei bereits, so müssen Sie hier nochmals bestätigen, um wirklich zu speichern.

Wie bei allen Requesterfenstern im FileMonitor, bedeutet auch hier das Drücken einer beliebigen Taste einen Abbruch.

Falls Sie während der Benutzung des FileMonitor einmal ein CLI-Fenster benötigen, können Sie vom Projektmenü aus eines öffnen. Der RUN- und der NEWCLI-Befehl müssen sich dazu jedoch in Ihrem CLI-Directory befinden. Vom Projektmenü aus kann man den FileMonitor auch beenden. Dies ist auch durch Betätigung des Close-Gadgets machbar.

Beispiele

Um die Funktionen des FileMonitor zu überprüfen, sehen Sie in Abbildung 1 einige Beispiele anhand von Systemprogrammen. Aber wie schon gesagt, bitte nur an Kopien experimentieren, damit im Falle eines Falles die Originale noch vorhanden sind.

Bei allen Patches, bei denen Texte ersetzt werden, ist darauf zu achten, daß der Ersatztext die Länge des alten Textes nicht überschreiten darf. Ist der Ersatztext kürzer, so empfiehlt es sich, die restlichen Zeichen durch Leerzeichen zu überschreiben. Eventuell hilft es auch, eine '0' an das Ende des Ersatztextes zu hängen. Wer dies nicht beachtet, wird beim Starten des gepatchten Programms wahrscheinlich sehr bald indischen Besuch bekommen. (Viele Grüße vom Guru!) Ansonsten viel Spaß mit Ihrem neuen FileMonitor!

(jb)

Programmname	Suchtyp	Suche nach	Ersetze durch	Funktion
AmigaBasic	HEX	323C00C8	323C00FF	volle Fenstergröße
C:NewCLI	ASCII	0/0/640/100	0/0/640/256	" "
C:Setclock	HEX	4EAEFF22	4E714E71	DATE-Befehl nicht ausführen (spart Zeit)
GenAm2	HEX	028000BC	028000F4	angenehme Fenstergröße

Abb. 1. Einige Beispiele, um Bytes gezielt zu ändern

Listings

```
*********************************************
*** Filemonitor V2.0                 **************
*** written in 1989 by HochiSoft **************
*** (c) 1990 by DMV-Verlag           **************
*** Sprache : Assembler (DEVPAC) **************
*********************************************

 opt o+,ow-
 *opt d+
ARP  equ 1

*** Include-Dateien laden
 incdir "RAD:include/","SYS:include/"    ;anpassen !!!
 include exec/exec_lib.i
 include exec/memory.i
 include libraries/dos_lib.i
 include libraries/dosextens.i
 include libraries/dos.i
 include intuition/intuition.i
 include intuition/intuition_lib.i
 include graphics/graphics_lib.i
```

Listing: FileMonitor.asm

```
 include devices/console_lib.i

** Konstanten aus arpbase.i
ArpVersion        equ 34                     ;Mindestversion
_LVOFileRequest   equ -$126
_LVOBaseName      equ -$276
_LVOTackOn        equ -$270
fr_Hail           equ 0
fr_File           equ 4
fr_Dir            equ 8
fr_SIZEOF         equ 30
**
FILENAMESIZE      equ 50
MWidth            equ 30*8+CHECKWIDTH
MHeight           equ 8
MLast             equ 1
MLeft             equ 5
MVY               equ 2
MItemNr           equ 19
Leiste1Count      equ 7
Leiste2Count      equ 2
LAST              equ -1
```

Listing: FileMonitor.asm

```
** WindowKonstanten **
WFlags          equ WINDOWDEPTH!ACTIVATE!WBENCHWINDOW!WINDOWCLOSE!WINDOWDRAG
IDCMP           equ REQSET!GADGETUP!MOUSEBUTTONS!CLOSEWINDOW!RAWKEY!MENUPICK

*** RawKeyCodes ***
F1_Taste        equ $50
F2_Taste        equ $51
F3_Taste        equ $52
F4_Taste        equ $53
F5_Taste        equ $54
F6_Taste        equ $55
F9_Taste        equ $58
F10_Taste       equ $59
O_Taste         equ $18
RETURN_Taste    equ $44
TAB_Taste       equ $42
EBSC_Taste      equ $45
L_Taste         equ $28
S_Taste         equ $21
Pfeil_Hoch      equ $4c
Pfeil_Runter    equ $4d
Pfeil_Links     equ $4f
Pfeil_Rechts    equ $4e

*** Variablen-Struktur ***
 rsreset
PageStart       rs.l    1
OldPageStart    rs.l    1
MyIntBase       rs.l    1
MyGfxBase       rs.l    1
MyDosBase       rs.l    1
MyArpBase       rs.l    1
ConsoleBase     rs.l    1
ReadFileHandle  rs.l    1
WriteFileHandle rs.l    1
FileBuffer      rs.l    1
MyWindow        rs.l    1
MyFont          rs.l    1
FileLoaded      rs.w    1
MyUserPort      rs.l    1
MyRastPort      rs.l    1
AsciiOn         rs.b    1
EditOn          rs.b    1
Changed         rs.b    1
HexByte         rs.b    1
HexOffs         rs.w    1
CrX             rs.w    1
CrY             rs.w    1
AltCx           rs.w    1
AltCy           rs.w    1
AltSecs         rs.l    1
AltMicros       rs.l    1
VarSize         rs.b    0

*** Makros ***

Lines   equ 20
TEXT    macro
\1Text
        dc.b    \2
        dc.b    0
\1SIZE  equ *-\1Text
        endm

* COMMAND F1_Taste,Laden
COMMAND         macro
     IFND       COMMCOUNT
COMMCOUNT   set 0
     ENDC
COMMCOUNT   set COMMCOUNT+1
        dc.w    \1
        dc.w    \2Routine-Routines
     endm

MENULEISTE  macro
     IFND    MENULEISTECOUNT
MENULEISTECOUNT set 0
     ENDC
MENULEISTECOUNT set MENULEISTECOUNT+1
        dc.b    \1SIZE*8
        endm


MENU            macro
     IFND       MENUCOUNT
MENUCOUNT       set 0
     ENDC
MENUCOUNT       set MENUCOUNT+1
        dc.w    %11111<<11!\1<<5!ActualMenu
        dc.w    \1Routine-Routines
        endm

LEISTE  macro
        dc.b    160,\1
        endm

LEISTCOM macro
        dc.w    \1Routine-Routines
        endm

* MItem Flags,ComKey,Last *,Text
MItem           macro
        dc.w    \1!ITEMTEXT!HIGHCOMP!ITEMENABLED
        dc.b    \2
        dc.b    \3
        *dc.w   \4Text-Datas
\4  equ ItemCount
ItemCount   set ItemCount+1
\4Menu      equ MNr
MNr     set MNr+1
        endm

CALLVEC     macro
        jsr _LVO\1(a6)
        endm

BytesPerLine     equ 6*4
BytesPerPage     equ BytesPerLine*Lines
TOP              equ 30
LEFT             equ 6
Status2Y         equ TOP+Lines*8+5
ReqHeight        equ 80
ReqWidth         equ 300
ReqY             equ 14
ReqLinks         equ LEFT
ReqX             equ (640-ReqWidth)/2
ReqRechts        equ 640-LEFT-ReqWidth
*** Memory-Struktur
        rsreset
Vars             rs.b    VarSize
FileName         rs.b    FILENAMESIZE
AsciiSearchString rs.b   FILENAMESIZE
HexSearchString  rs.b    FILENAMESIZE
SearchBuffer     rs.b    FILENAMESIZE
StelleAdresse    rs.b    10
RealFileName     rs.b    FILENAMESIZE*2
UndoBuffer       rs.b    FILENAMESIZE*2
WindowTitle      rs.b    80
ReqWindowTitle   rs.b    80
PagePuffer       rs.b    BytesPerPage
Line             rs.b    80
MyInputEvent     rs.b    24
IoRequest        rs.b    48
Puffer           rs.b    2
MenuLeistenStructs  rs.b    mu_SIZEOF*4
MenuStructs      rs.b    (it_SIZEOF+mi_SIZEOF)*MItemNr
WindowInfo       rs.b    nw_SIZE
MyStrGadget      rs.b    si_SIZEOF
MyBorder         rs.b    bd_SIZEOF
MyCursor         rs.b    it_SIZEOF
Dateizeigertext  rs.b    44
DosErrorText     rs.b    22
LInfoCoords1     rs.w    Leiste1Count+1
LInfoCoords2     rs.w    Leiste2Count+1
MyRequester      rs.b    rq_SIZEOF
ReqBorder        rs.b    bd_SIZEOF
MyGadget1        rs.b    gg_SIZEOF
MyGadget2        rs.b    gg_SIZEOF
MyGadget3        rs.b    gg_SIZEOF
MyGadget4        rs.b    gg_SIZEOF
MyBorder2        rs.b    bd_SIZEOF
MyBorder3        rs.b    bd_SIZEOF
MyBorder4        rs.b    bd_SIZEOF
MyText1          rs.b    it_SIZEOF
MyText2          rs.b    it_SIZEOF
MyText3          rs.b    it_SIZEOF
MyText4          rs.b    it_SIZEOF
ReqText1         rs.b    it_SIZEOF
ReqText2         rs.b    it_SIZEOF
ReqText3         rs.b    it_SIZEOF
CancelText       rs.b    it_SIZEOF
MemXY1           rs.w    10
MemXY2           rs.w    10
MemXY3           rs.w    10
MemXY4           rs.w    10

     IFD ARP
FileReq          rs.b    fr_SIZEOF
DirName          rs.b    FILENAMESIZE*2
     ENDC
GetMem           rs.b    0

Mem              equr    a4
Var              equr    Mem
FileSize         equr    a5

****** Programm ********

*** Workbench od. CLI ?
     move.l  d0,d7                          ;Parameter
     move.l  a0,a5                          ;retten
     and.w   #31,d7                         ;Parameterlaenge begrenzen
     move.l  (_SysBase).w,a6
     suba.l  a1,a1
     CALLVEC FindTask
     move.l  d0,a2
     tst.l   pr_CLI(a2)
     bne.s   Program
     lea     pr_MsgPort(a2),a0
     CALLVEC WaitPort
     lea     pr_MsgPort(a2),a0
     CALLVEC GetMsg
     move.l  d0,-(sp)
     moveq   #1,d7                          ;Parameterlaenge
     bsr.s   Program
     CALLVEC Forbid
     move.l  (sp)+,a1
     jmp     _LVOReplyMsg(a6)
```

Listing: FileMonitor.asm

```
Program
* Speicher holen
        move.l  #GetMem,d0
        move.l  #MEMF_PUBLIC!MEMF_CLEAR,d1
        CALLVEC AllocMem
        tst.l   d0
        beq     MemErr
        move.l  d0,Mem
* Strukturen initialisieren
        lea     MenuTexte(pc),a3
        lea     MenuLeisten(pc),a0
        lea     MenuLeistenStructs(Mem),a1
        bsr     GetMenuLeisten              ;Menus
        lea     MenuLeistenStructs(Mem),a0
        lea     MItem1(pc),a1
        lea.l   MenuStructs(Mem),a2
        bsr     GetMenu                     ;Menuitems
        lea     MyGadget1+4(Mem),a0 Gadget
        move.l  #14<<16+30,(a0)+
        move.l  #(ReqWidth-28)<<16+10,(a0)+
        addq.l  #2,a0
        move.l  #(RELVERIFY!ENDGADGET)<<16+STRGADGET!REQGADGET,(a0)+
        move.l  #49<<16+49,si_BufferPos+MyStrGadget(Mem)
        lea     UndoBuffer(Mem),a0
        move.l  a0,MyStrGadget+si_UndoBuffer(Mem)
        moveq   #1,d0
        lea     WindowInfo+4(Mem),a0        ;Window
        move.l  #640<<16+200,(a0)+
        move.w  d0,(a0)+
        move.l  #IDCMP,(a0)+
        move.l  #WFlags,(a0)+
        lea     WindowName(pc),a1
        move.l  a1,nw_Title+WindowInfo(Mem)
        move.w  d0,nw_Type+WindowInfo(Mem)
        lea     MyBorder(Mem),a0            ;Border
        move.l  #LEFT<<16+TOP,(a0)+
        move.l  #3<<24+$0205,(a0)+
        lea     XY(pc),a0                   ;XY
        lea     MemXY1(Mem),a1
        moveq   #4*10-1,d0
InitXYLoop
        move.b  (a0)+,d1
        ext.w   d1
        move.w  d1,(a1)+
        dbf     d0,InitXYLoop
        lea     MyCursor(Mem),a0            ;IntuiText
        move.l  #$03000600,(a0)+
        move.l  #LEFT<<16+TOP-6,(a0)+
        lea     Dateizeigertext(Mem),a1
        lea     ZeigerText(pc),a0
CopyZeigerText
        move.b  (a0)+,d0
        move.b  d0,(a1)+
        addq.b  #2,d0                       ;-2?
        bne.s   CopyZeigerText
        lea     MemXY4(Mem),a0
        lea     MyBorder2(Mem),a1
        moveq   #2,d1
InitBorder
        move.l  BBa0,bd_XY(a1)
        move.l  #$03000005,bd_FrontPen(a1)
        lea     bd_SIZEOF(a1),a1
        dbf     d1,InitBorder
        lea     MyBorder4(Mem),a1
        move.l  a1,MyBorder3+bd_NextBorder(Mem)
        lea     MemXY3(Mem),a0
        move.l  a0,bd_XY(a1)
        moveq   #2,d0
        lea     ReqText1(Mem),a0
        move.l  #14<<16+20,d1
InitCoord
        move.b  d0,(a0)
        move.l  d1,it_LeftEdge(a0)
        lea     it_SIZEOF(a0),a0
        subq.w  #8,d1
        bpl.s   InitCoord
* dos.library oeffnen
        IFD ARP
        lea     ArpNm(pc),a1
        moveq   #ArpVersion,d0
        CALLVEC OpenLibrary
        move.l  d0,MyArpBase(Mem)
        ENDC
        lea     DosName(pc),a1
        CALLVEC OldOpenLibrary
        move.l  d0,MyDosBase(Var)
* intuition.library oeffnen
        lea     IntName(pc),a1
        moveq   #33,d0                      ;Kick 1.2
        CALLVEC OpenLibrary
        move.l  d0,MyIntBase(Var)
        beq     IntErr
* graphics.library oeffnen
        lea     GfxName(pc),a1
        CALLVEC OldOpenLibrary
        move.l  d0,MyGfxBase(Var)
* Window oeffnen
        move.l  MyIntBase(Var),a6
        lea     WindowInfo(Mem),a0
        CALLVEC OpenWindow
        move.l  d0,MyWindow(Var)
        beq     WindowErr
        move.l  d0,a2
        lea     MenuLeistenStructs(Mem),a1
        move.l  a2,a0
```

Listing: FileMonitor.asm

```
        CALLVEC SetMenuStrip
        move.l  wd_UserPort(a2),MyUserPort(Var)
        move.l  wd_RPort(a2),MyRastPort(Var)
        move.l  wd_WScreen(a2),a0
        cmp.w   #212,sc_Height(a0)
        blo.s   NSize
        move.l  a2,a0
        moveq   #0,d0
        moveq   #12,d1
        CALLVEC MoveWindow
NSize
* Topaz.Font (8) holen
        move.l  MyGfxBase(Var),a6
        lea     MyTextAttr(pc),a0
        CALLVEC OpenFont
        move.l  d0,MyFont(Var)
        move.l  MyRastPort(Var),a1
        move.l  d0,a0
* Und setzen
        CALLVEC SetFont
        moveq   #3,d0
        bsr     Pen
        moveq   #TOP-10,d1
        bsr     DrawHori
        lea     GadgetLeiste1+1(pc),a3
        moveq   #TOP-12,d3
        move.w  #638-Leiste1Size*8+8,d0
        moveq   #Leiste1Size-1,d1
        bsr     Textout3
        lea     GadgetLeiste2(pc),a3
        move.w  #Status2Y,d3
        move.w  #638-Leiste2Size*8,d0
        moveq   #Leiste2Size,d1
        bsr     Textout3
        moveq   #10,d1
        lea     GadgetLeiste1+Leiste1Size(pc),a0
        lea     LInfoCoords1+Leiste1Count*2(Mem),a1
        moveq   #Leiste1Count-1,d0
        bsr     GetLeiste
        bsr     DrawHori
        move.w  #TOP+Lines*8-3,d1
        lea     GadgetLeiste2+Leiste2Size(pc),a0
        lea     LInfoCoords2+Leiste2Count*2(Mem),a1
        moveq   #Leiste2Count,d0
        bsr     GetLeiste
        bsr     DrawHori
        move.w  #198,d1
        bsr     DrawHori
* console - library oeffnen
        lea     ConsoleName(pc),a0
        lea     IoRequest(Mem),a1
        moveq   #-1,d0
        moveq   #0,d1
        move.l  (_SysBase).w,a6
        CALLVEC OpenDevice
        tst.l   d0
        bne     DeviceError

        move.l  20+IoRequest(Mem),ConsoleBase(Var)

        subq.w  #2,d7
        bmi.s   DoLoad
        cmp.b   #10,(a5)
        beq.s   DoLoad
        lea     RealFileName(Mem),a0
CopyParam
        move.b  (a5)+,(a0)+
        dbf     d7,CopyParam
        clr.b   (a0)
        bsr     LOADRoutine2
        bra.s   HoleMsg
DoLoad  bsr     DoBorder
        moveq   #0,d6
        bsr     LOADRoutine
*** Hauptschleife ***
HoleMsg
* Auf Message warten

        move.l  (_SysBase).w,a6
        move.l  MyUserPort(Var),a0
        CALLVEC WaitPort
        move.l  MyUserPort(Var),a0
        CALLVEC GetMsg
        move.l  d0,a0
        move.l  a0,a1
        lea     20(a0),a0
* Wichtige Werte retten
        move.l  (a0)+,d4        class
        move.w  (a0)+,d5        code
        move.w  (a0),d6         qual
        addq.l  #6,a0
        move.w  (a0)+,d7        mousex
        swap    d7
        move.w  (a0)+,d7        mousey
        move.l  (a0)+,d2        Secs
        move.l  (a0),d3         Micros
* InputEvent rekonstruieren
        cmp.w   #RAWKEY,d4
        bne.s   NCon2ie
        move.b  #1,4+MyInputEvent(Mem)   RAWKEY
        move.w  d5,MyInputEvent+6(Mem)   CODE
        move.w  d6,MyInputEvent+8(Mem)   Qual
NCon2ie
* Message beantworten
        CALLVEC ReplyMsg
        cmp.w   #CLOSEWINDOW,d4
```

Listing: FileMonitor.asm

```
        beq      Quit
        cmp.w    #MOUSEBUTTONS,d4
        bne      NKlick
        cmp.w    #SELECTDOWN,d5
GoNE_NextMsg
        bne.s    HoleMsg
        tst.b    FileLoaded(Var)
        beq.s    NL2
        cmp.w    #TOP-10,d7
        bhi.s    NL1
        lea      L1Info(pc),a1
        lea      LInfoCoords1(Mem),a0
        swap     d7
        bsr      TestLeiste
        bra      ClearHexBuf
NL1 cmp.w    #TOP+Lines*8-3,d7
        blo.s    NL2
        swap     d7
        lea      LInfoCoords2(Mem),a0
        lea      L2Info(pc),a1
        bsr      TestLeiste
        bra      ClearHexBuf
NL2
        move.w   d7,d1
        swap     d7
        sub.w    #TOP-6,d1
        blo      NLocate
        cmp.w    #Lines*8-1,d1
        bhi.s    NLocate
        move.w   d7,d0
        sub.w    #LEFT+8*(BytesPerLine*2+BytesPerLine/4),d0
        blo.s    NAscii
        lsr.w    #3,d0
        st.b     d7
DoLocate

        lsr.w    #3,d1
        cmp.w    #BytesPerLine-1,d0
        bhi.s    GoNE_NextMsg
        move.l   AltSecs(Var),d4
        move.l   AltMicros(Var),d5
        move.l   d2,AltSecs(Var)
        move.l   d3,AltMicros(Var)
        cmp.w    AltCx(Var),d0
        bne.s    DoLocate2
        cmp.w    AltCy(Var),d1
        bne.s    DoLocate2
        move.l   d4,d0
        move.l   d5,d1
        move.l   MyIntBase(Var),a6
        CALLVEC  DoubleClick
        tst.b    d0
        beq.s    Go_NextMsg
        bsr      SetEditMode
Go_NextMsg
        bra      NextMsg
DoLocate2

        bsr      CursorOff
        move.w   d0,CrX(Var)
        move.w   d1,CrY(Var)
        bsr      CursorOn
        bra.s    ClearHexBuf
NAscii           move.w           d7,d0
        sub.w    #LEFT-4,d0
        blo.s    NLocate
        lsr.w    #3,d0
        move.w   d0,d6
        ext.l    d6
        divu.w   #9,d6
        sub.w    d6,d0
        swap     d6
        tst.w    d6
        beq.s    NLocate
        subq.w   #1,d0
        lsr.w    #1,d0
        sf.b     d7
        bra.s    DoLocate

NAtHex
NLocate
NKlick
        cmp.w    #MENUPICK,d4
        bne.s    NoMenu
        lea      Menus(pc),a0
        moveq    #MENUCOUNT-1,d0
SearchMenu
        cmp.w    (a0),d5
        addq.l   #4,a0
        dbeq     d0,SearchMenu
        bne.s    Go_NextMsg
        bra.s    StartComm
NoMenu
* Auf Funktion der Taste ueberpruefen
        cmp.w    #RAWKEY,d4
        bne      NoRawKey
        tst.b    FileLoaded(Var)
        beq      NoRawKey
        lea      Commands(pc),a0
        moveq    #COMMCOUNT-1,d0
SearchComm
        cmp.w    (a0),d5
        addq.l   #4,a0
        dbeq     d0,SearchComm
        bne.s    NoSub
StartComm
        lea      Routines(pc),a1
        add.w    -(a0),a1
        jsr      (a1)                          ;Funktion anspringen
ClearHexBuf
        pea      NextMsg(pc)
        bra      ClearHexBufSub
NoSub
        move.w   CrX(Var),d2
        bsr      CheckLimits
        beq      NoRawKey
* Editieren
        tst.b    EditOn(Var)
        beq      NoRawKey
        move.l   ConsoleBase(Var),a6
        lea      MyInputEvent(Mem),a0
        lea      Puffer(Mem),a1
        suba.l   a2,a2
        moveq    #1,d1
        CALLVEC  RawKeyConvert                 ;Asciiwert holen
        tst.w    d0
        ble      NoRawKey
        move.b   Puffer(Mem),d5
        cmp.b    #32,d5
        blo      NoRawKey
        move.w   CrY(Var),-(sp)
        move.w   CrX(Var),-(sp)
        move.l   PageStart(Var),-(sp)
        tst.b    AsciiOn(Var)
        bne.s    NHexEd
* Hex-Seite edieren
        bsr      UpperD5
        sub.b    #'0',d5
        bmi.s    CharOverFlow
        cmp.b    #9,d5
        bls.s    NGreaterDez
        subq.b   #'A'-'9'-1,d5
NGreaterDez
        cmp.b    #$F,d5
        bhi.s    CharOverFlow
        move.w   HexOffs(Var),d3
        subq.w   #1,HexOffs(Var)
        lsl.w    #2,d3
        and.b    #$F,d5
        lsl.w    d3,d5
        or.b     d5,HexByte(Var)
        move.b   HexByte(Var),d5
        tst.w    d3
        bne.s    CharOverFlow
        pea      NHexEd(pc)
ClearHexBufSub
        move.w   #1,HexOffs(Var)
        clr.b    HexByte(Var)
        rts
CharOverFlow
        move.l   (sp)+,a0
        movem.w  (sp)+,d0-d1
        bra.s    CharOverFlow2
NHexEd
        bsr      CRghtRoutine
        bsr      CursorOn
        move.l   (sp)+,a0
        cmpa.l   PageStart(Var),a0 gescrollt?
        beq.s    NAdaptY
        subq.w   #1,2(sp)
NAdaptY
        movem.w  (sp),d0-d1
        bsr      PageOffset2
        move.l   PageStart(Var),a2
        move.b   d5,(a2,d0)
        st.b     Changed(Mem)
* aktuelle Zeile neu anzeigen
ShowLine
        adda.w   d1,a2
        movem.w  (sp)+,d0-d1
        move.l   MyGfxBase(Var),a6
        move.w   d1,d0
        lsl.w    #3,d0
        add.w    #TOP,d0
        bsr      Linelocate
        bsr      ConvertLine
        bsr      DoBorder
CharOverFlow2
NoRawKey
NextMsg
        bra      HoleMsg
Quit
* Datei schliessen, Dateispeicher freigeben
        bsr      LooseChangesTest
        beq.s    NextMsg
        bsr      CloseDatei
        move.l   (_SysBase).w,a6
        lea      IoRequest(Mem),a1
        CALLVEC  CloseDevice
DeviceError
* Window weg!
        move.l   MyFont(Var),a1
        move.l   MyGfxBase(Var),a6
        CALLVEC  CloseFont
        move.l   MyIntBase(Var),a6
        move.l   MyWindow(Var),a0
        CALLVEC  ClearMenuStrip
        move.l   MyWindow(Var),a0
        CALLVEC  CloseWindow
WindowErr
* GFX weg
```

Listing: FileMonitor.asm

```
        move.l  (_SysBase).w,a6
        move.l  MyGfxBase(Var),a1
        CALLVEC CloseLibrary
* Intuition schliessen
        move.l  MyIntBase(Var),a1
        CALLVEC CloseLibrary
IntErr
* No more DOS!
        move.l  MyDosBase(Var),a1
        CALLVEC CloseLibrary
        IFD ARP
        move.l  MyArpBase(Mem),d0
        beq.s   NArpClose
        move.l  d0,a1
        CALLVEC CloseLibrary
NArpClose
        ENDC
* Free the Mem!
        move.l  Mem,a1
        move.l  #GetMem,d0
        CALLVEC FreeMem
        moveq   #0,d0
MemErr          rts                             ;Hello EXEC, I'm
  ready!


*** Tasten-Routinen ***
Routines
*** Cursor Hoch ***
CUpRoutine
        bsr.s   AltTest
        bne     FILESTARTRoutine
        bsr.s   ShiftTest
        beq.s   NShift
PAGEUPRoutine
        move.l  PageStart(Var),a2
        suba.w  #BytesPerPage,a2
        move.w  CrX(Var),d2
        ext.l   d2
        cmpa.l  FileBuffer(Var),a2
        blo     FileLocate
        bsr     CursorOff
        move.l  a2,PageStart(Var)
        bsr     ZeigeSeite
        bra.s   ReturnSub7
```

Listing: FileMonitor.asm

```
NShift
CUp2Routine
        tst.w   CrY(Var)
        bne.s   NScrUp
        bsr     CursorOff
        moveq   #-BytesPerLine,d2
        bsr     CheckLimits
        beq.s   NUp
        moveq   #-8,d1
        moveq   #TOP,d0
        bsr     Scroll

        move.l  PageStart(Var),a2
        adda    d2,a2
        move.l  a2,PageStart(Var)
        bsr     ConvertLine
        bsr     PufDown
NUp
        bsr     CursorOn
        bra.s   ReturnSub7
NScrUp
        subq.w  #1,CrY(Var)
        bra     Cursor

* In: Qualifier in d6, Out: Z=1 wenn Alt *
AltTest         move.b  d6,d0
                and.b   #%110000,d0
ReturnSub7
        rts

* In: Qualifier in d6, Out: Z=1 wenn Shift *
ShiftTest
        move.b  d6,d0
        and.b   #%11,d0
        rts

*** Cursor unten ***
CDownRoutine
        bsr.s   AltTest
        bne     FILEENDRoutine
        bsr.s   ShiftTest
        beq.s   NShiftDown
PAGEDOWNRoutine
        move.l  PageStart(Var),a2
        adda.w  #BytesPerPage,a2
```

Listing: FileMonitor.asm

```
        move.l  a2,a1
        *add.w  #BytesPerPage,a1
        suba.l  FileBuffer(Var),a1
        cmpa.l  FileSize,a1
        bhi     FILEENDRoutine
        bsr     CursorOff
        move.l  a2,PageStart(Var)
        bsr     ZeigeSeite
        bra.s   NDown2
NShiftDown
CDown2Routine
        cmp.w   #Lines-1,CrY(Var)
        bne.s   NScrDown
        bsr     CursorOff
        move.w  #BytesPerLine,d2
        bsr.s   CheckLimits
        beq.s   NDown
        move.w  #-BytesPerPage,d2
        bsr.s   CheckLimits
        moveq   #8,d1
        move.w  #TOP+(Lines-1)*8,d0
        bsr     Scroll

        move.l  PageStart(Var),a2
        lea     BytesPerPage(a2),a2
        bsr     ConvertLine

        lea     -BytesPerPage(a2),a2
        move.l  a2,PageStart(Var)
        bsr     PufUp
NDown
        bsr.s   CursorOn
        bra.s   NDown2
NScrDown
        addq.w  #1,CrY(Var)
        bsr.s   Cursor
NDown2
        rts
*** Cursor Links ***
CLftRoutine
        subq.w  #1,CrX(Var)
        *tst.w  CrX(Var)
        bpl.s   Cursor
        move.w  #BytesPerLine-1,CrX(Var)
        bra     CUp2Routine

*** Cursor Rechts ***
CRghtRoutine
        addq.w  #1,CrX(Var)
        cmp.w   #BytesPerLine,CrX(Var)
        bne.s   Cursor
        clr.w   CrX(Var)
        bra     CDown2Routine

QUITRoutine
        addq.l  #4,sp
        bra     Quit

** d2=Offset zu altem Wert * Z=1 wenn Error **
CheckLimits
        move.l  PageStart(Var),a2
        bsr.s   PageOffset
        add.w   d2,d0
        adda.w  d0,a2
        move.l  FileBuffer(Var),a0
        cmpa.l  a0,a2
        bmi.s   OverFlow
        move.l  FileSize,a1
        adda.w  CrX(Var),a1
        adda.l  a1,a0
        cmp.l   a0,a2
        blo.s   ExitCheck
OverFlow
        moveq   #0,d0
ExitCheck
        rts

* aus CrX u. CrY Offset zum ersten Byte der Seite nach D
0
PageOffset
        move.w  CrX(Var),d0
        move.w  CrY(Var),d1
PageOffset2
        mulu    #BytesPerLine,d1
        add.w   d1,d0
        ext.l   d0
        rts


*** Cursor an/aus Routine
Cursor
        bsr.s   DoBorder
CursorOn
        move.l  PageStart(Var),OldPageStart(Var)
        move.w  CrX(Var),AltCx(Var)
        move.w  CrY(Var),AltCy(Var)
        movem.l d0-d1/a1/a0/a6,-(sp)
        lea     FilePosText-ZeigerText+Dateizeigertext(Mem),
a0
        bsr.s   PageOffset
        move.l  PageStart(Var),d1
        add.l   d1,d0
        sub.l   FileBuffer(Var),d0
        bsr     LONGToHEX
        move.l  MyGfxBase(Var),a6
        move.l  MyRastPort(Var),a1
        moveq   #5+(dateizeigende-dateizeig)*8,d0
        move.w  #Status2Y,d1
        CALLVEC Move
        bsr     Pen1
        move.l  MyRastPort(Var),a1
        lea     FilePosText-ZeigerText+Dateizeigertext(Mem),
a0
        moveq   #8,d0
        CALLVEC Text
        movem.l (sp)+,d0-d1/a1/a0/a6
CursorOff
DoBorder
        movem.l d0-d2/a0-a2/a6,-(sp)
        move.w  AltCx(Var),d0
        bsr.s   GetHexx
        lea     MemXY1(Mem),a2
        move.l  MyIntBase(Var),a6
        bsr.s   DoDrawBorder
        lea     MemXY2(Mem),a2
        move.w  AltCx(Var),d0
        lsl.w   #3,d0
        add.w   #6*9*8,d0
        bsr.s   DoDrawBorder
        bsr.s   EditCursor
        movem.l (sp)+,d0-d2/a0-a2/a6
        rts

DoDrawBorder
        move.w  AltCy(Var),d1
        lsl.w   #3,d1
        lea     MyBorder(Mem),a1
        move.l  a2,bd_XY(a1)
        move.l  MyRastPort(Var),a0
        jmp     _LVODrawBorder(a6)

GetHexx         move.w          d0,d2
        lsl.w   #1,d0
        *ext.l  d2
        *divu.w #4,d2
        lsr.w   #2,d2
        add.w   d2,d0
        lsl.w   #3,d0
        rts


* Scrolle Ausschnitt vertikal um D1
Scroll
        move.l  MyGfxBase(Var),a6
        movem.l d0-d5/a1,-(sp)
        moveq   #0,d0
        move.l  MyRastPort(Var),a1
        moveq   #LEFT,d2
        moveq   #TOP-6,d3
        move.w  #LEFT+78*8,d4
        move.w  #TOP+(Lines-1)*8+1,d5
        CALLVEC ScrollRaster
        movem.l (sp)+,d0-d5/a1

* GFX-Cursor an LEFT,Y=d0 setzen
Linelocate
        move.w  d0,d1
        move.l  MyRastPort(Var),a1
        moveq   #LEFT,d0
        jmp     _LVOMove(a6)

*** EditCursor an/aus Routinen
EditCursor2
        move.w  CrX(Var),d0
        move.w  CrY(Var),d1
        bra.s   EditCursor3
EditCursor
        tst.b   EditOn(Var)
        beq.s   NoEditCur
        move.w  AltCy(Var),d1
        move.w  AltCx(Var),d0
EditCursor3
        lsl.w   #3,d1
        tst.b   AsciiOn(Var)
        beq.s   HexOn
        lea     Space(pc),a0
        lsl.w   #3,d0
        add.w   #(BytesPerLine*2+BytesPerLine/4)*8,d0
        bra.s   EditCurOut
HexOn
        lea     DoubleSpace(pc),a0
        bsr.s   GetHexx

EditCurOut
        lea     MyCursor(Mem),a1
        move.l  a0,it_IText(a1)
        move.l  MyRastPort(Var),a0
        move.l  MyIntBase(Var),a6
        CALLVEC PrintIText
NoEditCur
ReturnSub10
        rts

*** Neue Datei laden
LOADRoutine
        bsr     LooseChangesTest
        beq.s   ReturnSub10
        bsr     CursorOff
        lea     LadenFrage(pc),a0
        clr.b   16+FileReq(Mem)
        bsr     GetFileName
        beq     NichtLaden2
```

Listing: FileMonitor.asm

```
LOADRoutine2
    bsr      CloseDatei
    lea      RealFileName(Mem),a0
    move.l   a0,d1
    move.l   MyDosBase(Var),a6
    move.l   #MODE_OLDFILE,d2
    CALLVEC  Open
    move.l   d0,ReadFileHandle(Var)
    bhi.s    Load
    bsr      IO_Error_Out
    bra.s    NichtLaden
Load
    bsr GetSize
    move.l   (_SysBase).w,a6
    move.l   FileSize,d0
    add.l    #BytesPerPage,d0
    move.l   #$10000,d1
    CALLVEC  AllocMem
    move.l   d0,FileBuffer(Var)
    bne.s    MemOk
    lea      LowMemText(pc),a3
    moveq    #1,d0
    bsr      MessageWindow
    bra.s    NichtLaden
MemOk
    move.l   FileBuffer(Var),a0
    adda.l   FileSize,a0
    move.w   #BytesPerPage-1,d1
EndCharLoop
    move.b   #"#",(a0)+
    dbf      d1,EndCharLoop
    move.l   FileBuffer(Var),a2
    move.l   a2,PageStart(Var)
    bsr      LoadBuffer
    move.l   d0,-(sp)
    move.l   ReadFileHandle(Var),d1
    CALLVEC  Close
    cmp.l    (sp)+,FileSize
    beq.s    NLoadError
    bsr      CloseDatei
NLoadError
NichtLaden
    bsr.s    ShowTitle
    lea      FileLenText-ZeigerText+Dateizeigertext(Mem),
 a0
    move.l   FileSize,d0
    bsr      LONGToHEX
NichtLaden2
    bsr      Pen1
    move.l   MyWindow(Var),a2
    lea      Dateizeigertext(Mem),a3
    move.w   #Status2Y,d3
    moveq    #ZeigerTextEnde-ZeigerText,d1
    move.l   MyGfxBase(Var),a6
    bsr      Textout2
    bra      ZeigeSeiteImmer

******** WindowTitel zeigen! ********
ShowTitle
    lea      WindowTitle(Mem),a0
    move.l   a0,a1
    lea      RealFileName(Mem),a2
    moveq    #78/2-1,d0
CopyTitle
    move.w   (a2)+,(a0)+
    dbf      d0,CopyTitle
    move.l   MyWindow(Var),a0
ShowTitle2
    tst.l    FileBuffer(Var)
    bne.s    ShowFileName
    lea      WindowName(pc),a1
ShowFileName
    lea.l    ScreenTitle(pc),a2
    move.l   MyIntBase(Var),a6
    jmp      _LVOSetWindowTitles(a6)



*** Datei speichern ***
SAVERoutine
    tst.b    FileLoaded(Var)
    bne.s    FileIsThere
    lea      NoFileText(pc),a3
    moveq    #0,d0
    bra      MessageWindow
FileIsThere
    lea      SpeichernFrage(pc),a0
    bset     #5,16+FileReq(Mem)
    bsr      GetFileName
    beq.s    Nichtschreiben
    lea      RealFileName(Mem),a2
    move.l   a2,d1
    move.l   a2,d4
    move.l   #ACCESS_READ,d2
    move.l   MyDosBase(Var),a6
    CALLVEC  Lock
    move.l   d0,d1
    beq.s    NoOldFile
    CALLVEC  UnLock
    lea      SaveVerifyText(pc),a3
    bsr      YesNoRequest
    beq.s    ReturnSub8
NoOldFile
    move.l   d4,d1
    move.l   #MODE_NEWFILE,d2
    CALLVEC  Open
```

Listing: FileMonitor.asm

```
    move.l   d0,WriteFileHandle(Var)
    bhi.s    Save
    bsr      IO_Error_Out
    bra.s    Nichtschreiben
Save
    move.l   FileBuffer(Var),a0
    move.l   a0,d2
    move.l   FileSize,d3
    move.l   WriteFileHandle(Var),d1
    CALLVEC  Write
    move.l   d0,d5
    bpl.s    CloseIt
    bsr      IO_Error_Out
    bsr.s    CloseIt
    move.l   d4,d1
    jmp      _LVODeleteFile(a6)

CloseIt
    move.l   WriteFileHandle(Var),d1
    CALLVEC  Close
    tst.l    d5
    bmi.s    ReturnSub8
    bsr      ShowTitle
    sf.b     Changed(Var)
Nichtschreiben
ReturnSub8
    rts

WriteError


*** Seitenpuffer zurueckschreiben
RESTORERoutine
    tst.b    FileLoaded(Var)
    beq.s    ReturnSub8
    bsr      CopyToPage
    bsr      CursorOff
    bra      ZeigeSeiteImmer


*** AsciiString suchen
ASCIIPREVRoutine
    moveq    #-1,d7
    bra.s    StartSearch
ASCIIFINDRoutine
    lea      ASCII_suchText(pc),a0
    move.w   #ReqRechts,d2
    bsr      GetString2
    bhi.s    ASCIIPREVRoutine
    beq.s    ReturnSub8
ASCIINEXTRoutine
    moveq    #1,d7
StartSearch
    tst.b    AsciiSearchString(Mem)
    beq.s    ReturnSub8
    bsr      CursorOff
    lea      AsciiSearchString(Var),a1
    moveq    #0,d4    ;laenge
AsciilenLoop
    tst.b    (a1,d4)
    beq.s    LenFound
    addq.l   #1,d4
    bra.s    AsciilenLoop
LenFound
    tst.b    d7
    bpl.s    n
    neg.l    d4
    subq.l   #1,a1
    sub.l    d4,a1
n
    move.l   MenuLeistenStructs+1*mu_SIZEOF+mu_FirstItem(
Mem),a0
    btst     #0,mi_Flags(a0)
    seq.b    d3
SEARCH
    bsr      PageOffset
    ext.l    d0
    move.l   PageStart(Var),a0
    add.l    a0,d0
    move.l   FileBuffer(Var),a0
    sub.l    a0,d0
    moveq    #0,d1
    tst.b    d7
    bpl.s    n3
    sub.l    d4,d0
    cmp.l    FileSize,d0
    bls.s    ngt
    move.l   FilnSize,d0
ngt
    bra.s    AsciiSearch
n3
    addq.l   #1,d0    ;erstes ueberspringen
AsciiSearch
    sub.l    d1,d0
    move.l   d7,d1
    neg.l    d1
AsciiSearch2
    cmp.l    FileSize,d0
    bhi      NoLocate
    add.l    d7,d1
    cmp.w    d4,d1
    beq.s    ASCIIFound
    add.l    d7,d0
    bmi      NoLocate
    move.b   -1(a0,d0.l),d2
    move.b   (a1,d1.l),d5
```

Listing: FileMonitor.asm

```
        tst.b   d3
        bne.s   CaseIsAn
        cmp.b   #"a",d2
        blo.s   NUpper
        bclr    #5,d2
NUpper          bsr.s   UpperD5
CaseIsAn
        cmp.b   d5,d2
        bne.s   AsciiSearch
        bra.s   AsciiSearch2
ASCIIFound
        sub.l   d1,d0
        tst.b   d7
        bpl.s   n2
        add.l   d1,d0
        subq.l  #1,d0
n2 move.l       d0,d2
        bra     FileLocate2

UpperD5
        cmp.b   #"a",d5
        blo.s   NUpper2
        bclr    #5,d5
ReturnSub9
NUpper2         rts

* HexString suchen
HEXPREVRoutine
        moveq   #-1,d7
        bra.s   HEXStart
HEXFINDRoutine
        lea     HEX_suchText(pc),a0
        moveq   #ReqLinks,d2
        bsr     GetString3
        beq.s   ReturnSub9
        bhi.s   HEXPREVRoutine
HEXNEXTRoutine
        moveq   #1,d7
HEXStart
        tst.b   HexSearchString(Mem)
        beq.s   ReturnSub9
        bsr     CursorOff
        lea     HexSearchString(Mem),a2
        lea     SearchBuffer(Mem),a1
        moveq   #-1,d4  laenge
ConvfromHex
        addq.l  #1,d4
        moveq   #0,d2   ergebnisByte
        moveq   #4,d3   shiftwert
GetHexByte
        tst.w   d3
        bmi.s   ConvfromHex
        move.b  (a2)+,d5
        beq.s   Convready
        bsr.s   UpperD5 UpperCase
        sub.b   #'0',d5
        bmi.s   HexStringErr
        cmp.b   #9,d5
        bls.s   NGreaterDez2
        subq.b  #'A'-'9'-1,d5
NGreaterDez2
        cmp.b   #$F,d5
        bhi.s   HexStringErr
        lsl.w   d3,d5
        or.b    d5,d2
        subq.l  #4,d3
        move.b  d2,(a1,d4.l)
        bra.s   GetHexByte

Convready
        tst.b   d7
        bpl.s   n4
        neg.l   d4
        subq.l  #1,a1
        sub.l   d4,a1
n4
        cmp.w   #4,d3
        beq     SEARCH
HexStringErr
        lea.l   HEX_ErrText(pc),a3
        moveq   #0,d0
        bsr MessageWindow
Go_ZeigeSeite
        bra     ZeigeSeite

FILESTARTRoutine
        moveq   #0,d2
        bra.s   FileLocate


FILEENDRoutine
        move.l  FileSize,d2
        subq.l  #1,d2

FileLocate
        bsr     CursorOff
FileLocate2
        move.l  FileBuffer(Var),a0
        cmp.l   FileSize,d2
        bhs.s   NoLocate
        move.l  d2,d0
        divu.w  #BytesPerLine,d0
        bvs.s   NoLocate
        clr.w   CrY(Var)
        swap    d0
        move.w  d0,CrX(Var)
```

Listing: FileMonitor.asm

```
        ext.l   d0
        sub.l   d0,d2
        lea     (a0,d2.l),a0
        move.l  a0,PageStart(Var)
NoLocate

        bra.s   Go_ZeigeSeite

STELLERoutine
        lea     MyStrGadget+si_MaxChars(Mem),a1
        move.w  #9,(a1)
        lea     StelleFrage(pc),a0
        move.w  #ReqX,d2
        bsr     GetString5
        move.w  #49,(a1)
        tst.b   d0
        beq.s   ReturnSub5
        bsr     CursorOff
        lea     StelleAdresse(Mem),a0
        moveq   #0,d2
GetOffsLoop
        move.b  (a0)+,d1
        beq.s   ReadyOffs
        cmp.b   #'9',d1
        bls.s   NGreaterDez3
        bclr    #5,d1
        subq.b  #'A'-'9'-1,d1
NGreaterDez3
        sub.b   #'0',d1
        bmi.s   HexStringErr
        cmp.b   #$F,d1
        bhi     HexStringErr
        lsl.l   #4,d2
        or.b    d1,d2
        bra.s   GetOffsLoop
ReadyOffs
        cmp.l   FileSize,d2
        blo.s   FileLocate2
        bsr     CursorOn
        bra     FILEENDRoutine


** Ganze Datei einlesen
LoadBuffer
        move.l  MyDosBase(Var),a6
        move.l  FileBuffer(Var),a0
        move.l  a0,d2
        move.l  FileSize,d3
        move.l  ReadFileHandle(Var),d1
        beq.s   NotLoad
        CALLVEC Read
        st.b    FileLoaded(Var)
ReturnSub5
NotLoad         rts

EditHEXRoutine
        clr.b   d7
        bra.s   SetEditMode
EditASCIIRoutine
        st.b    d7
SetEditMode
        tst.b   EditOn(Var)
        beq.s   MustEditMode
        cmp.b   AsciiOn(Var),d7
        beq.s   ReturnSub5
MustEditMode
        bsr.s   EDITMODERoutine
        bra.s   SetEditMode

* Editmode wechseln
EditmodeRoutine
EDITMODERoutine
        tst.b   EditOn(Var)
        bne.s   NStartEdit
EditmodeRoutine2
        bsr.s   EditStartRoutine
NStartEdit
        bsr.s   Go_EditCursor2
        Not.b   AsciiOn(Var)
        bra.s   Go_EditCursor2

EditAusRoutine
        tst.b   EditOn(Var)
        beq.s   ReturnSub5
EditStartRoutine
        Not.b   EditOn(Var)
Go_EditCursor2
        bra     EditCursor2

* Dateimemory freigeben
CloseDatei
        move.l  (_SysBase).w,a6
        move.l  FileBuffer(Var),d0
        beq.s   NoFreeBuf
        clr.l   FileBuffer(Var)
        move.l  d0,a1
        move.l  FileSize,d0
        add.l   #BytesPerPage,d0
        CALLVEC FreeMem
NoFreeBuf
        sf.b    Changed(Var)
        sf.b    FileLoaded(Var)
        clr.l   PageStart(Var)
        suba.l  FileSize,FileSize
        rts
```

Listing: FileMonitor.asm

```

** Neues CLI-Fenster oeffnen
NEWCLIRoutine
        move.l  MyDosBase(Var),a6
        lea     EmptyStream(pc),a0
        move.l  a0,d1
        move.l  #MODE_NEWFILE,d2
        CALLVEC Open
        move.l  d0,d2
        move.l  d0,d3
        beq.s   ReturnSub5
        lea     NewCliCom(pc),a0
        move.l  a0,d1
        CALLVEC Execute     CLI-Befehl starten
        move.l  d3,d1
        jmp     _LVOClose(a6)

** Von / In Seitenpuffer kopieren
CopyFromPage
        move.l  PageStart(Var),a0
        lea     PagePuffer(Mem),a1
        bra.s   CopyPage

CopyToPage
        move.l  PageStart(Var),a1
        lea     PagePuffer(Mem),a0

CopyPage
        move.w  #BytesPerPage/4-1,d0
copyPageLoop
        move.l  (a0)+,(a1)+
        dbf     d0,copyPageLoop
        rts

** Seitenpuffer - Scrollroutinen
PufDown
        lea.l   PagePuffer(Mem),a0
        move.w  #(BytesPerPage-BytesPerLine)-1,d0
PufdLoop
        move.b  (a0,d0.w),BytesPerLine(a0,d0.w)
        dbf     d0,PufdLoop
        lea     PagePuffer(Mem),a0
        move.l  PageStart(Var),a1
        moveq   #BytesPerLine/4-1,d0
Pufdadd
```

Listing: FileMonitor.asm

```
        move.l  (a1)+,(a0)+
        dbf     d0,Pufdadd
        rts

PufUp
        lea.l   PagePuffer(Mem),a0
        move.w  #(BytesPerPage-BytesPerLine)/4-1,d0
PufupLoop
        move.l  BytesPerLine(a0),(a0)+
        dbf     d0,PufupLoop
        lea     PagePuffer+BytesPerPage-BytesPerLine(Mem),a0
        move.l  PageStart(Var),a1
        lea     BytesPerPage-BytesPerLine(a1),a1
        moveq   #BytesPerLine/4-1,d0
Pufupadd
        move.l  (a1)+,(a0)+
        dbf     d0,Pufupadd
        rts


* Seitenanzeige neu aufbauen
ZeigeSeiteImmer
        bset    #1,OldPageStart+3(Var)  *unmoeglicher PageStart!
ZeigeSeite
        move.l  a2,-(sp)
        move.l  PageStart(Var),a2
        cmp.l   OldPageStart(Var),a2
        beq.s   NZeige
        moveq   #TOP,d1
        moveq   #Lines-1,d2
SeiteLoop
        movem.l a0/a1/a6/d0-d1,-(sp)
        move.l  MyRastPort(Var),a1
        move.l  MyGfxBase(Var),a6
        moveq   #LEFT,d0
        CALLVEC Move
        bsr.s   ConvertLine
        movem.l (sp)+,a0/a1/a6/d0-d1
        addq.w  #8,d1
        lea     6*4(a1),a1
        dbf     d2,SeiteLoop
NZeige  bsr     CursorOn
        move.l  (sp)+,a2
        bra     CopyFromPage
```

Listing: FileMonitor.asm

```
Pen1            moveq   #1,d0
* aktuellen Farbstift aendern
Pen             movem.l a0-a2/a6/d1,-(sp)
    move.l  MyGfxBase(Var),a6
    move.l  MyRastPort(Var),a1
    CALLVEC SetAPen
    movem.l (sp)+,a0-a2/a6/d1
    rts


* Zeile in Zeilenpuffer hex/ascii schreiben
* a2=erstes Byte
ConvertLine
    movem.l a0-a1/d0-d1,-(sp)
    moveq   #5,d1
    lea     Line(Mem),a0
    tst.b   FileLoaded(Var)
    bne.s   ConvLineLoop
    moveq   #77,d0
ClearLine
    move.b  #" ",(a0)+
    dbf     d0,ClearLine
    bra.s   NoConv
ConvLineLoop
    move.l  (a2)+,d0
    bsr.s   LONGToHEX
    addq.l  #8,a0
    move.b  #32,(a0)+
    dbf     d1,ConvLineLoop
    lea     -6*4(a2),a2
    moveq   #6*4-1,d1
ConvLineLoop2
    move.b  (a2)+,d0
    cmp.b   #32,d0
    bhs.s   Nopoint
    moveq   #".",d0
Nopoint
    move.b  d0,(a0)+
    dbf     d1,ConvLineLoop2
NoConv
    move.l  MyRastPort(Var),a1
    lea     Line(Mem),a0
    moveq   #78,d0
    CALLVEC Text
    movem.l (sp)+,a0-a1/d0-d1
    rts

*** d0 = INT, a0=Zieladresse *** keine Register veraendert ***
LONGToHEX
    movem.l d0-d2/a0,-(sp)
    moveq   #32/4-1,d1
ToHEXLoop
    rol.l   #4,d0
    move.b  d0,d2
    lsl.b   #4,d2
    lsr.b   #4,d2
    move.b  d2,(a0)
    cmp.b   #$a,d2
    blo.s   Lower10
    addq.b  #'A'-'9'-1,(a0)
Lower10     add.b   #'0',(a0)+
    dbra    d1,ToHEXLoop
    movem.l (sp)+,d0-d2/a0
    rts

* Dateilaenge ermitteln
GetSize
    movem.l d0-d3/a0-a2/a6,-(sp)
    move.l  MyDosBase(Var),a6
    move.l  ReadFileHandle(Var),d1
    moveq   #0,d2
    moveq.l #OFFSET_END,d3
    CALLVEC Seek
    move.l  ReadFileHandle(Var),d1
    moveq   #0,d2
    moveq.l #OFFSET_BEGINNING,d3
    CALLVEC Seek
    move.l  d0,FileSize
    movem.l (sp)+,d0-d3/a0-a2/a6
    rts



GetFileName
    IFD ARP
    lea     RealFileName(Mem),a1
    bsr     ShiftTest
    bne     GetFileName2
    move.l  MyArpBase(Mem),d0
    beq     GetFileName2
    move.l  d0,a6
    move.l  a0,-(sp)
    lea     DirName(Mem),a0
CopyDir     move.b  (a1)+,(a0)+
    bne.s   CopyDir
    lea     DirName(Mem),a0
    CALLVEC BaseName
    move.l  d0,a2
    lea     FileName(Mem),a1
    move.l  a1,a0
CopyName
    move.b  (a2)+,(a0)+
    bne.s   CopyName
    move.l  d0,a2
    clr.b   (a2)
    lea     FileReq(Mem),a0
    move.l  a0,a2
    move.l  (sp)+,(a2)+
    move.l  a1,(a2)+
    lea     DirName(Mem),a1
    move.l  a1,(a2)+
    move.l  MyWindow(Var),(a2)+
    CALLVEC FileRequest
    tst.l   d0
    beq.s   ExitGetName
    lea     RealFileName(Mem),a2
    move.l  a2,a0
    lea     DirName(Mem),a1
CopyDirName
    move.b  (a1)+,(a2)+
    bne.s   CopyDirName
    lea     FileName(Mem),a1
    CALLVEC TackOn
    moveq   #1,d0       Z-Flag loeschen
    ENDC
ExitGetName
    rts

* Öffnet ein Fenster, schreibt Textseite in A0, fordert Eingabe *
GetString5
    movem.l a1-a6/d2-d4,-(sp)
    lea     StelleAdresse(Mem),a1
    moveq   #2,d0
Go_GetString4
    moveq   #1,d1
    bra.s   GetString4
GetString3
    movem.l a1-a6/d2-d4,-(sp)
    lea     HexSearchString(Mem),a1
    moveq   #3,d0
    bra.s   Go_GetString4
GetString2
    movem.l a1-a6/d2-d4,-(sp)
    moveq   #3,d0
    moveq   #0,d1
    lea     AsciiSearchString(Mem),a1
    bra.s   GetString4


GetFileName2
    moveq   #2,d0
    moveq   #0,d1
    move.w  #ReqX,d2
    movem.l a1-a6/d2-d4,-(sp)
GetString4
    move.w  d2,MyRequester+rq_LeftEdge(Mem)
    move.w  d0,d2
    move.l  a1,MyStrGadget(Mem)
    clr.w   MyStrGadget+si_BufferPos(Mem)
    clr.w   MyStrGadget+si_DispPos(Mem)
    lea     ReqText1(Mem),a5
    move.l  a0,it_IText(a5)
    move.l  d1,it_NextText(a5)
    beq.s   NSecondText
    lea     it_SIZEOF(a5),a1
    move.l  a1,it_NextText(a5)
    move.l  a0,it_IText(a1)
    clr.l   it_NextText(a1)
GetTNext
    tst.b   (a0)+
    bne.s   GetTNext
    move.l  a0,it_IText(a5)
NSecondText
    lea     MyRequester(Mem),a0
    lea     6(a0),a1
    move.w  #ReqY,(a1)+
    move.l  #ReqWidth<<16+ReqHeight,(a1)+
    clr.l   (a1)+
    lea     MyGadget1(Mem),a2
    move.l  a2,(a1)+
    lea     MyStrGadget(Mem),a3
    move.l  a3,gg_SpecialInfo(a2)
    lea     MyGadget2(Mem),a3
    move.l  a3,(a2)
    movem.l a0/a1/a5,-(sp)
    lea     GadgetTexte(pc),a1
    lea     MyText1(Mem),a0
    moveq   #ReqWidth>>2,d1
    lsl.w   #2,d1
    divu    d0,d1
    move.w  d1,d0
    sub.w   #7*8,d0
    lsr.w   #1,d0
    move.w  d2,d3
    moveq   #0,d3
    subq.w  #3,d0
GadgetInit
    clr.l   (a3)
    move.l  a3,-gg_SIZEOF(a3)
    addq.l  #4,a3
    move.w  d0,(a3)+
    add.w   d1,d0
    move.w  #ReqHeight-22,(a3)+ gg_TopEdge
    move.l  #(7*8+9)<<16+8+9,(a3)+
    addq.l  #2,a3
    move.l  #(RELVERIFY+ENDGADGET)<<16+REQGADGET!BOOLGADGET,(a3)+
    lea MyBorder2(Mem),a5
    move.l  a5,(a3)+    gg_GadgetRender
```

Listing: FileMonitor.asm

```
        addq.l  #4,a3
        move.l  a0,(a3)+     gg_IntuiText
        move.w  #$0300,(a0) it_FrontPen
        move.l  #$00050005,it_LeftEdge(a0)
        move.l  a1,it_IText(a0) Textadresse
        addq.l  #8,a1
        lea     it_SIZEOF(a0),a0
        addq.l  #8,a3
        move.w  d3,(a3)+     gg_GadgetID
        addq.l  #4,a3
        addq.w  #1,d3
        cmp.w   d2,d3
        bne.s   GadgetInit
        lea     MyBorder3(Mem),a0
        move.l  a0,MyGadget2+gg_GadgetRender(Mem)
        movem.l (sp)+,a0/a1/a5
        lea     ReqBorder(Mem),a2
        move.l  #$02020005,bd_FrontPen(a2)
        lea     XY5(pc),a3
        move.l  a3,bd_XY(a2)
        move.l  a2,(a1)+
        move.l  a5,(a1)+
        move.b  #1,rq_BackFill(a0)
        move.l  MyWindow(Var),a1
        move.l  MyIntBase(Var),a6
        CALLVEC Request
        tst.b   d0
        beq.s   RequestErr
        move.l  #REQSET,d3
        bsr.s   MsgWait
        move.l  MyIntBase(Var),a6
        lea     MyGadget1(Mem),a0
        move.l  MyWindow(Var),a1
        lea     MyRequester(Mem),a2
        CALLVEC ActivateGadget
        move.l  #GADGETUP,d3
        bsr.s   MsgWait
        move.l  MyStrGadget(Mem),a0
        move.b  (a0),d0
        beq.s   ExitRequest
        move.w  gg_GadgetID(a2),d0
        subq.w  #1,d0   -0+ ID
ExitRequest
        movem.l (sp)+,a1-a6/d2-d4
        rts
RequestErr
        lea     NoReqText(pc),a3
        moveq   #0,d0
        bsr     MessageWindow
        bra.s   ExitRequest



MsgWait     move.l  MyUserPort(Var),a0
        move.l  (_SysBase).w,a6
        CALLVEC WaitPort
        move.l  MyUserPort(Var),a0
        CALLVEC GetMsg
        move.l  d0,a1
        move.l  im_Class(a1),d2
        move.l  im_IAddress(a1),a2
        CALLVEC ReplyMsg
        cmp.l   d3,d2
        bne.s   MsgWait
        rts


*** Unterrroutine fuer HelpWindow und RequesterWindow ( a6=gfx,a2=Win.,a3=Text) ****
Textout2                    *** d3=y, wie Textout
        moveq   #5,d0           x
Textout3
        move.w  d1,-(sp)
        move.w  d3,d1           y => d1
        addq.w  #8,d3           y=Y+8
        move.l  wd_RPort(a2),a1     rastport
        CALLVEC Move            ...
        move.w  (sp)+,d0
        move.l  a3,a0           Textadr fuer Textroutine setzen
        move.l  wd_RPort(a2),a1     rastport
        CALLVEC Text            raus damit
        rts

*** Ausgabe von  DOS-Fehlermeldungen, a6=dos
IO_Error_Out
        movem.l a0-a3/d0-d2,-(sp)
        CALLVEC IoErr
        lea     DosErrorText+IoErrorString-IoErrorText(Mem),a1
        moveq   #100,d2
ConvToDec
        divu    d2,d0
        move.b  d0,(a1)
        add.b   #'0',(a1)+
        swap    d0
        ext.l   d0
        divu    #10,d2
        bne.s   ConvToDec
        lea     DosErrorText(Mem),a3
        moveq   #0,d0
        bsr.s   MessageWindow
        moveq   #0,d1           Wenn, IO_Error_Out aufgerufen, dann war Ladefehler
        movem.l (sp)+,a0-a3/d0-d2
NoErr
ReturnSub11
        rts


LooseChangesTest
        move.b  Changed(Var),d0
        not.b   d0
        bne.s   ReturnSub11
        lea     WarnungsText(pc),a3
YesNoRequest
        movem.l a0-a6/d0-d7,-(sp)
        lea     MyText1(Mem),a2
        move.w  #$0200,(a2)
        move.l  #$00050003,it_LeftEdge(a2)
        lea     OkText(pc),a0
        move.l  a0,it_IText(a2)
        moveq   #0,d0
        bra.s   MessageWeiter
* Gibt einen Requester mit Infos fuer den User aus, wartet auf ReturnTaste *
MessageWindow
        movem.l a0-a6/d0-d7,-(sp)
        suba.l  a2,a2
MessageWeiter
        lea     ReqText3(Mem),a1
        move.l  a3,it_IText(a1)
        lea     -it_SIZEOF(a1),a5
        move.l  d0,it_NextText(a1)
        beq.s   NDreiTexte
        lea     -it_SIZEOF(a5),a0
        move.l  a5,it_NextText(a1)
        move.l  a0,it_NextText(a5)
Loop1   tst.b   (a3)+
        bne.s   Loop1
        move.l  a3,it_IText(a5)
Loop2   tst.b   (a3)+
        bne.s   Loop2
        move.l  a3,it_IText(a0)
NDreiTexte
        lea     CancelText(Mem),a3
        move.w  #$0200,(a3)
        move.l  #$00050003,it_LeftEdge(a3)
        lea     AbbruchText(pc),a0
        move.l  a0,it_IText(a3)
        suba.l  a0,a0
        moveq.l #0,d0
        move.l  #VANILLAKEY,d1
        move.w  #ReqWidth+ReqWidth/4,d2
        moveq   #ReqHeight*2/3-4,d3
        move.l  MyIntBase(Var),a6
        CALLVEC AutoRequest
        move.l  a2,d1
        beq.s   ExitReq
        tst.b   d0
ExitReq     movem.l (sp)+,a0-a6/d0-d7
        rts

* d1=y
DrawHori
        move.l  MyGfxBase(Var),a6
        move.l  MyRastPort(Var),a1
        moveq   #2,d0
        movem.l a1/d1,-(sp)
        CALLVEC Move
        movem.l (sp)+,a1/d1
        move.w  #637,d0
        jmp     _LVODraw(a6)

* d0=x d1=starty
DrawVerti
        move.l  MyGfxBase(Var),a6
        move.l  MyRastPort(Var),a1
        movem.l a1/d0/d1,-(sp)
        CALLVEC Move
        movem.l (sp)+,a1/d0/d1
        add.w   #10,d1
        jmp     _LVODraw(a6)

* d7=MouseX Leiste=a0 LeistenRoutinen=a1 LCount=d0
TestLeiste
        move.w  (a1)+,d0
        cmp.w   (a0)+,d7
        blo.s   ExitL
        cmp.w   (a0),d7
        bhi.s   TestLeiste
        lea     Routines(pc),a1
        add.w   d0,a1
        move.l  a1,-(sp)
        jsr     (a1)
        move.l  MyDosBase(Var),a6
        moveq   #8,d1
        CALLVEC Delay
        bra.s   MsgTest
LeisteCall
        move.l  (sp),a1
        jsr     (a1)        Funktion anspringen
MsgTest     move.l  (_SysBase).w,a6
        move.l  MyUserPort(Var),a0
        CALLVEC GetMsg
        tst.l   d0
        beq.s   LeisteCall
        move.l  d0,a1
        CALLVEC ReplyMsg
        move.l  (sp)+,a1
ExitL       rts
```

Listing: FileMonitor.asm

```
* d0=Count d1=Y  a0=LeisteEnde  a1=LeisteInfoCoords
GetLeiste
    moveq    #0,d2
    bra.s    SetLeiste
GetLeisteLoop
    addq.w   #1,d2
    cmp.b    #160,-(a0)
    bne.s    GetLeisteLoop
    subq.l   #2,a1
SetLeiste
    movem.l  a0-a1/d0/d1,-(sp)
    move.w   d2,d0
    lsl.w    #3,d0
    add.w    #-639,d0
    neg.w    d0
    move.w   d0,(a1)
    bsr.s    DrawVerti
    movem.l  (sp)+,a0-a1/d0/d1
    subq.w   #1,d0
    bpl.s    GetLeisteLoop
    rts

* a0=Menuleiste, a1=Definitionen
GetMenu
    move.l   a0,-(sp)
    move.l   a2,mu_FirstItem(A0)
    moveq    #0,d1         ;y
GetMenuLoop
    lea      it_SIZEOF+mi_SIZEOF(a2),a0
    move.b   3(a1),d0
    bpl.s    NextFound
    suba.l   a0,a0
NextFound
    move.l   a0,(a2)+      NextItem
    clr.w    (a2)+         LeftEdge
    move.w   d1,(a2)+      TopEdge
    move.w   #MWidth-MLeft,(a2)+ Width
    move.w   #MHeight,(a2)+  Height
    move.w   (a1)+,(a2)+ Flags
    clr.l    (a2)+         MutualExcl.
    btst.b   #0,-1(a1)
    beq.s    NCheck
    move.b   (a1),-1(a2)
NCheck lea mi_SIZEOF-mi_ItemFill(a2),a0
    move.l   a0,(a2)+      ItemFill
    clr.l    (a2)+         SelectFill
    move.b   (a1)+,(a2)+ Commkey
    addq.l   #1,a1
    addq.l   #1,a2
    clr.l    (a2)+         SubItem
    clr.w    (a2)+         NextSelect
* IntuiText erstellen
    move.w   #1<<7+0,(a2)+   Pen/Paper
    clr.w    (a2)+         DrawMode/Even
    move.w   #CHECKWIDTH,(a2)+ LeftEdge
    clr.w    (a2)+   TopEdge
    clr.l    (a2)+   Font
    move.l   a3,(a2)+      String
NextTextLoop2
    tst.b    (a3)+
    bne.s    NextTextLoop2
    clr.l    (a2)+         Next
    add.w    #MHeight+3,d1
    tst.b    d0
    bmi.s    ExitMenuLoop
    add.b    d0,d1
    bra.s    GetMenuLoop
ExitMenuLoop
    move.l   (sp)+,a0
    move.l   mu_NextMenu(a0),d1
    move.l   d1,a0
    bne.s    GetMenu
ExitGetMenu
    rts

* a0= Definitionen a1=Strukturen a3=Texte
GetMenuLeisten
    moveq    #MLeft,d1
    moveq    #0,d0
GetMenuLeistenLoop
    move.b   (a0)+,d0
    beq.s    ExitGetMenuLeisten
    lea      mu_SIZEOF(a1),a2
    move.l   a2,(a1)+
    move.w   d1,(a1)+
    add.w    d0,d1
    clr.w    (a1)+
    move.w   d0,(a1)+
    move.l   #MHeight<<16+MENUENABLED,(a1)+
    move.l   a3,(a1)+
NextTextLoop1
    tst.b    (a3)+
    bne.s    NextTextLoop1
    move.l   a2,a1
    bra.s    GetMenuLeistenLoop
ExitGetMenuLeisten
    clr.l    -mu_SIZEOF(a2)
    rts

Datas
DosName      DOSNAME
IntName      INTNAME
GfxName      GRAFNAME
ArpNm            dc.b     "arp.library",0
```

Listing: FileMonitor.asm

```
FontName       dc.b       "topaz.font",0
WindowName     dc.b       "*** Filemonitor V2.0 *** - "
NoFileText     dc.b       "Keine Datei geladen",0
ScreenTitle    dc.b       "FILEMONITOR V2.0  written in 1989
  by HOCHISOFT - "
               dc.b       169,"1990 DMV-Verlag",0
ConsoleName    dc.b       "console.device",0
NewCliCom      dc.b       "NEWCLI >"
EmptyStream    dc.b       "NIL:",0
    even
MyTextAttr     dc.l       FontName
               dc.w       8
               dc.b       FS_NORMAL
               dc.b       FPF_ROMFONT

XY             dc.b       -1,-7,8+9-1,-7,8+9-1,2,-1,2,-1,-6
XY2            dc.b       -1,-7,8,-7,8,2,-1,2,-1,-6
XY3            dc.b       2,2,7*8+7,2
               dc.b       7*8+7,8+6,2,8+6,2,2
XY4            dc.b       0,0,7*8+9,0
               dc.b       7*8+9,8+8,0,8+8,0,0
    even
XY5            dc.w       2,1,ReqWidth-3,1
               dc.w       ReqWidth-3,ReqHeight-2,2,ReqHeight
-2
               dc.w       2,1


MItem1
ItemCount      set 0
MNr            set 0
    MItem      COMMSEQ,<"L">,0,LOAD
    MItem      COMMSEQ,<"S">,0,SAVE
    MItem      0,0,MVY,NEWCLI
    MItem      COMMSEQ,<"Q">,LAST,QUIT
ItemCount      set 0
    MItem      CHECKIT!CHECKED,%10,0,CASESENSOFF
    MItem      CHECKIT,%1,MVY,CASESENSON
    MItem      COMMSEQ,<"F">,0,ASCIIFIND
    MItem      COMMSEQ,<"N">,0,ASCIINEXT
    MItem      COMMSEQ,<"P">,MVY,ASCIIPREV
    MItem      COMMSEQ,<"H">,0,HEXFIND
    MItem      COMMSEQ,<"W">,0,HEXNEXT
    MItem      COMMSEQ,<"R">,LAST,HEXPREV
ItemCount      set 0
    MItem      COMMSEQ,<"T">,0,FILESTART
    MItem      COMMSEQ,<"B">,MVY,FILEEND
    MItem      COMMSEQ,<"G">,LAST,STELLE
ItemCount      set 0
    MItem      COMMSEQ,<"-">,0,EditAus
    MItem      COMMSEQ,<"X">,0,EditHEX
    MItem      COMMSEQ,<"A">,MVY,EditASCII
    MItem      COMMSEQ,<"U">,LAST,RESTORE

MenuLeisten
    MENULEISTE  MenuLeiste1
    MENULEISTE  MenuLeiste2
    MENULEISTE  MenuLeiste3
    MENULEISTE  MenuLeiste4
               dc.b     0
               even

Menus
ActualMenu    set 0
    MENU     LOAD
    MENU     SAVE
    MENU     NEWCLI
    MENU     QUIT
ActualMenu set 1
    MENU     ASCIIFIND
    MENU     ASCIINEXT
    MENU     ASCIIPREV
    MENU     HEXFIND
    MENU     HEXPREV
    MENU     HEXNEXT
ActualMenu set 2
    MENU     FILESTART
    MENU     FILEEND
    MENU     STELLE
ActualMenu set 3
    MENU     EditAus
    MENU     EditHEX
    MENU     EditASCII
    MENU     RESTORE

Commands
    COMMAND Pfeil_Hoch,CUp
    COMMAND Pfeil_Runter,CDown
    COMMAND Pfeil_Links,CLft
    COMMAND Pfeil_Rechts,CRght
    COMMAND TAB_Taste,Editmode
    COMMAND F10_Taste,EditStart

GadgetTexte
OkText            dc.b     "  Ok!  ",0
AbbruchText
                  dc.b     "Abbruch",0
                  dc.b     "Vorige!",0

ZeigerText
dateizeig         dc.b     " Dateizeiger: "
dateizeigende
FilePosText    dc.b       "00000000  Dateilaenge: "
FileLenText    dc.b       "00000000"
ZeigerTextEnde
```

Listing: FileMonitor.asm

```
IoErrorText    dc.b    "AmigaDos Fehler "
IoErrorString ds.b    3
               dc.b    0,-2
   even
DoubleSpace    dc.b    " "
Space          dc.b    " ",0

MenuTexte
   TEXT    MenuLeiste1,<"Projekt  ">
   TEXT    MenuLeiste2,<"Suchen  ">
   TEXT    MenuLeiste3,<"Position  ">
   TEXT    MenuLeiste4,<"Edieren  ">

**** Projekt
   TEXT    LOAD,<"Laden">
   TEXT    SAVE,<"Speichern">
   TEXT    NEWCLI,<"CLI-Fenster oeffnen">
   TEXT    QUIT,<"Ende">
**** Suchen
   TEXT    CASESENSOFF,<"Gross/Klein gleich">
   TEXT    CASESENSON,<"Gross/klein verschieden">
   TEXT    ASCIIFIND,<"String suchen">
   TEXT    ASCIINEXT,<"naechsten String suchen">
   TEXT    ASCIIPREV,<"vorigen String suchen">
   TEXT    HEXFIND,<"Hexfolge suchen">
   TEXT    HEXNEXT,<"naechste Hexfolge suchen">
   TEXT    HEXPREV,<"vorige Hexfolge suchen">
**** Position
   TEXT    FILESTART,<"Dateianfang">
   TEXT    FILEEND,<"Dateiende">
   TEXT    STELLE,<"HEX-Offset eingeben">
**** Edieren
   TEXT    EditAus,<"Edieren aus">
   TEXT    EditHEX,<"HEX-Haelfte edieren">
   TEXT    EditASCII,<"ASCII-Haelfte edieren">
   TEXT    RESTORE,<"Seiten-Undo">

GadgetLeiste1
   LEISTE  <"Dateistart ">
   LEISTE  <"~ende ">
   LEISTE  <"Ed HEX ">
   LEISTE  <"Ed aus ">
   LEISTE  <"Ed ASCII ">
   LEISTE  <"Seite rueckwaerts ">
```

Listing: FileMonitor.asm

```
   LEISTE  <"Zeile rueckwaerts ">
Leiste1Size     equ *-GadgetLeiste1
GadgetLeiste2
   LEISTE  <"Seite vorwaerts  ">
   LEISTE  <"Zeile vorwaerts  ">
Leiste2Size     equ *-GadgetLeiste2

   even
L1Info
   LEISTCOM     FILESTART
   LEISTCOM     FILEEND
   LEISTCOM     EditHEX
   LEISTCOM     EditAus
   LEISTCOM     EditASCII
   LEISTCOM     PAGEUP
   LEISTCOM     CUp2
L1InfoEnde
L2Info LEISTCOM     PAGEDOWN
   LEISTCOM     CDown2
L2InfoEnde

LadenFrage      dc.b    "Welche Datei wollen Sie laden?"
     ,0
SpeichernFrage dc.b    "Unter welchem Namen speichern?"
     ,0
StelleFrage     dc.b    "Bei welchem HEXOffset (z.B.'A2b
     ')",0
                dc.b    "positionieren ?",0
HEX_suchText    dc.b    "Bitte geben Sie die zu suchende
     ",0
                dc.b    "Hex-Folge ein (z.B. '2e6C'):",0
HEX_ErrText     dc.b    "Fehlerhafte HEXZahl!",0
ASCII_suchText dc.b    "Welche Zeichenfolge suchen Sie?
     ",0
LowMemText      dc.b    "Nicht genuegend Speicher fuer P
     uffer!",0
                dc.b    "Bitte andere Tasks entfernen",0
                dc.b    "und wieder versuchen.",0
NoReqText       dc.b    "Zu wenig Speicher fuer Requeste
     r!",0
WarnungsText    dc.b    "Aenderungen verlieren?",0
SaveVerifyText dc.b    "Alte Datei wirklich ueberschrei
     ben?",0
```

Listing: FileMonitor.asm



Liebe Leser,
in dieser Rubrik finden Sie Rat und Hilfestellung zu Ihren kleineren und größeren Programmierproblemen und Informationen zu den in der AMIGA DOS vorgestellten Programmen und Produkten. Wir sind natürlich jederzeit bemüht, die eingehenden Leserfragen zu beantworten. Doch haben Sie bitte Verständnis, daß wir nicht alle eingehenden Briefe persönlich beantworten können. Oft erreichen uns mehrere Briefe zum gleichen Thema, einer davon wird dann stellvertretend für alle in unserer Zeitschrift beantwortet.
Dringliche Probleme, die das Heft betreffen, lassen sich womöglich besser telefonisch regeln. Rufen Sie dienstags unsere Hotline an. Von 17-20h stehen wir Ihnen mit Rat und Tat zur Seite, wenn Sie eine der folgenden Nummern wählen:
05651/809-740 (bis 744) *Ihre AMIGA-DOS-Redaktion*

SIE FRAGEN, WIR ANTWORTEN!

Probleme mit Fish 187 – DiskPerf

Ich habe einige Probleme mit dem PD-Programm DiskPerf auf der Fish-Disk 187. Das Programm dient dazu, die Datenübertragungs-Geschwindigkeit einer Festplatte zu messen. Wenn ich das Programm mit dem Aufruf "Diskperf DH0:" starte, führt es ein Directory-Scan und einen Seek-Read-Test durch. Danach erscheint die Fehlermeldung "Could not get 524288 Bytes of Memory". Obwohl 1 MByte zur Verfügung steht, mangelt es angeblich an Speicherplatz.

Meine Frage: Fehlen beim Start des Programms irgendwelche Parameter, oder muß ich noch andere Dinge beachten?

M. Sihn, Mühlacker

Das Problem liegt offensichtlich daran, daß Ihrem Rechner nicht genügend Speicher zur Verfügung steht, zumindest nicht genug "durchgängiger Speicher". Wahrscheinlich verwenden Sie noch eine alte DiskPerf-Version. Es sind bereits einige Updates zu diesem Programm erschienen, auf die Sie zurückgreifen sollten. Diverse PD-Vertreiber werden Ihnen sicherlich weiterhelfen können.

(Red.)

Adresse gesucht

Auf der Messe AMIGA '89 wurde ein Umbausatz für die 500er-Modelle vorgestellt, der aus einem Gehäuse besteht, in dem die alte Platine nur eingeschraubt wird und durch ein Kabel mit Erweiterungssteckplätzen verbunden wird. Für Laufwerke sollen wie beim 2000er Laufwerksschächte vorhanden sein. Das Gehäuse ist mit einem stärkeren Netzteil versehen, damit die Spannungsversorgung für Erweiterungen gewährleistet ist. Leider weiß ich nicht, welche Firma diesen Bausatz anbietet. Könnt Ihr mir die Adresse mitteilen?

C. Stelter, Wilhelmshaven

Das System heißt MW 550 und wird von der Firma Computer & Zubehör Miky Wenngatz angeboten. Die vollständige Adresse lautet:

Computer & Zubehör
Miky Wenngatz
Jägerweg 31
8031 Gilching
Tel.: 08105/24540
Fax: 08105/24530

Info zum BSG9-Virus

Der Virus nimmt grundsätzlich den ersten Befehl, der in der Startup-Sequenz vorkommt, an. Ein Beispiel: Steht als erstes der Befehl SetPatch, so kopiert der BSG9-Virus diesen Befehl in das Devs-Directory (jedoch ohne Namen aber mit Datums- und Größenangabe). Danach nimmt er den Namen SetPatch an und kopiert sich ins C-Verzeichnis. Nach Ausführung des Befehls in der Startup-Sequence, kopiert er sich ins RAM und leitet den Befehl in das Verzeichnis Dev um. Wird nun auf Diskette etwas abgespeichert, so überprüft BSG9 den ersten Eintrag in der Startup-Sequence. Ist dieser Eintrag zufällig AddBuffers, kopiert er diesen Befehl ohne Namen ins Devs-Directory und sich selbst unter dem Namen AddBuffers ins C-Verzeichnis. Ich hoffe, die Leser können mit dieser Information etwas anfangen.

D. Koritsch, Hamburg

Schreib/Lese-Fehler

Seit rund fünf Monaten bin ich stolzer Besitzer eines Amiga 500. Seit geraumer Zeit bereiten mir die Anzeigen von Read/Write Errors bei mehreren Disketten doch erhebliches Kopfzerbrechen. Zunächst schob ich diese Probleme auf die Lagerung der Disketten in der Nähe meiner Amateurfunkstation, bzw. in der Nähe des Netzteils. Jedoch stellte ich nach einiger Zeit des Testens fest, daß es hieran nicht liegen kann. Wodurch werden diese Diskettenfehler verursacht? Gibt es ein sicher arbeitendes und leicht zu bedienendes Programm, um solche Disketten zu reparieren?

H. Lange, Horb 1

Zu diesem Problem erreichten uns bereits mehrere Leseranfragen, die alle das gleiche Problem schilderten. Stellvertretend für diese Anfragen haben wir den Leserbrief von Herrn Lange veröffentlicht.

Da Sie dieses Problem bei mehreren Disketten festgestellt haben, deutet es auf einen Link-Virus hin. Hier tut sich besonders der Lamer Exterminator hervor, der – einmal auf Diskette vorhanden – diese gänzlich unleserlich macht. Genauer gesagt, auf der vom Lamer befallenen Diskette wird ein Datenblock mit dem Wort "Lamer!" beschrieben und damit gänzlich vom Virus verändert, so daß ein Read/Write Error die Folge ist.

Um sich vor solchen Virenangriffen schützen zu können, gehört ein gutes Virenschutzprogramm zur Grundausstattung eines jeden Amiga. Wir können Ihnen aus dem Bereich der Public Domain das Programm VirusX 4.0 empfehlen, das bei allen versierten PD-Anbietern im Angebot

ist. Sie können sich das Programm in Ihr C-Verzeichnis der Arbeitsworkbench kopieren und in der Startup-Sequence festhalten. Danach erscheint am oberen Bildschirmrand ein kleines Fenster, das Sie jederzeit aktivieren können, um eine neu eingelegte Diskette zu überprüfen. Das Programm erkennt eigenständig die geläufigsten Viren und kann bei Erkennen die verseuchte Diskette auch reparieren.

Um herauszufinden, wann sich ein Linkvirus in Ihrem System "breitgemacht" hat, können Sie auch unser Listing LinkChecker in dieser Ausgabe benutzen.

(Red.)

Probleme beim Seka Assembler

In der AMIGA DOS 2'90 wurde unter dem Titel "Fehler im 68000" auf Seite 112 auf ein merkwürdiges Phänomen im Zusammenhang mit dem Befehl "AND.L A1,D1" verwiesen. Zur Lösung des Problems:

Der Fehler liegt beim Assembler. Der Seka-Assembler mag zwar den Befehl codieren und später auch decodieren, die Befehls-Kombination ist aber im 68000er nicht zulässig. Der Seka steht damit aber nicht allein: auch der eine oder andere versiertere Assembler scheint vor solchen Reinfällen nicht gefeit. Daher auch ein kleines Lob an den PD-Assembler A68K (zum Beispiel Fish 186), der sich standhaft weigerte, diesen Befehl zu übersetzen und dabei folgende Meldung von sich gab: "Adressing mode not allowed here".

G. Steffens, Harsewinkel

Digi View Gold

In Ihrer Ausgabe vom Februar 1990 haben Sie das Programm Digi View Gold getestet. Da ich dieses Programm gerne erstehen würde, möchte ich Sie bitten, mir eine Adresse mitzuteilen, wo ich dieses Programm bekommen kann.

J. Ehrnsberger, Neumarkt

Das Programm Digi View Gold finden Sie normalerweise im gut sortierten Fachhandel. Da Sie mit dieser Aussage sicherlich nicht sehr viel anfangen können, hier einige Adressen. Digi View Gold ist zu beziehen bei:

GTI GmbH
Zimmersmühlenweg 73
6370 Oberursel
Tel: 06171/73048/9
Fax: 06171/8302

UNLIMITED
M. Hottenbacher
Kehrstr. 23
6200 Wiesbaden
Tel: 06121/543848

Creative Computers GmbH
Lutterothstr. 58
2000 Hamburg 20
Tel: 040/407332
Fax: 040/4919237

Atlantis GmbH
Dunantstr. 53
Postfach 1141
5030 Hürth
Tel: 02233/41081
Fax: 02233/46266

amigaOberland
Hohenwaldstr. 26
6374 Steinbach
Tel: 06171/71846
Fax: 06171/74805

(Red.)

Anschlußprobleme?

Seit einiger Zeit besitze ich neben dem C64 einen Amiga 500. Daß ich mit diesem neuen Gerät auch gern einen Drucker betreiben würde, versteht sich natürlich von selbst. Ich besitze jedoch einen STAR-LC-10-Color mit seriellem Kabel. Meine Frage dazu:

Kann ich diesen Drucker durch irgendeine Hardware etc. für meinen Amiga nutzen? Kann man den C64 eventuell als Interface benutzen? Welche Möglichkeiten gibt es dazu?

F. Heine, Wolfsburg

Verschiedene Firmen bieten Interfaces an, die Ihr Problem lösen dürften. Um den C64 jedoch als "Druckerpuffer" zu benutzen, bedarf es etwas mehr Hardwarepraxis.

Eine mögliche Konstellation sähe so aus: Verbindung des Amiga-Parallel-Port mit dem Userport des C64, der durch ein spezielles Programm (am besten in Assembler geschrieben, um Timing-Problemen aus dem Weg zu gehen) den Userport als parallelen Port verwaltet. Der RAM-Bereich könnte die empfangenen Daten auffangen und an den Drucker weitergeben. Verschiedentlich wurden in anderen Publikationen schon Bauanleitungen gegeben, einige Hardware-Bücher zum C64 enthalten ebenfalls Hinweise, hier ist allerdings ein Umschauen in der Elektronikbranche vonnöten.

(Red.)

Termine abfragen

Ich suche nach einer effektiven Möglichkeit, wichtige persönliche Daten und Termine beim Einschalten des Rechners abfragen zu können oder an diese erinnert zu werden. Gibt es eine Batch-Datei für die Startup-Sequence, oder gibt es ein kurzes Basic-Listing dafür?

B. Haidner, Wien

Sicher ist es möglich, sich ein Programm zu schreiben, das beim Einschalten des Rechners an wichtige Termine erinnern soll. Dazu bedarf es nur der Abfrage der (sofern vorhanden) batteriegepufferten Echtzeituhr im Amiga.

Wir können jedoch auf ein kommerzielles Produkt hinweisen, das wir im Rahmen der Aktuellseiten in Ausgabe 2/90 vorgestellt haben. Das Programm heißt Wer!Was! Wann!Wo!, dessen Demoversion frei weitergegeben werden darf.

(Red.)

Erweiterungen für Amiga 1000

Ich habe einen Amiga 1000, PAL-Version mit einer Erweiterung von 1MByte (3State). Dazu habe ich einige Fragen: Kann ich beim Amiga 1000 einen Resettaster einbauen, der auch das Kickstart 'rausschmeißt? Wenn ja, wie funktioniert das?

Wie sieht es mit Erweiterungskarten aus? Kann ich beim 1000er auch Turbokarten, PC-Karten und ähnliches einbauen, oder bin ich da auf spezielle Erweiterungen, die es nur für den Amiga 1000 gibt, angewiesen?

T. Hüsemann, Oberhausen

Rein theoretisch müßte ein Hardwarereset, der über den Expansionsport ausgeführt wird, auch den Kickstart löschen. Betreffs der Erweiterung für den Amiga 1000: PC- und Turbokarten können nicht in den Amiga 1000 eingebaut werden. Erweiterungen werden in der Regel an den Expansionsport angeschlossen. Um am Amiga 1000 auch MS-DOS benutzen zu können, muß ein Sidecar (Spezielle PC-Erweiterung für den Amiga 1000) angeschlossen werden.

(Red.)

DTP bzw. Musik mit dem Amiga

Ich möchte zwei Vereinszeitungen herausbringen, was mit den heutigen Computermöglichkeiten eigentlich kein Problem sein dürfte. Das Zauberwort heißt hier Desktop Publishing. Da ich weder Krösus noch Computerfreak bin, ist guter Rat teuer. Was benötige ich unbedingt an Hardware, damit ich überhaupt mit DTP arbeiten kann, und welche Software können Sie, wenn man das Preis/Leistungsverhältnis mit in Betracht zieht, empfehlen? Das zweite Thema, das mir unter den Nägeln brennt, ist Musik. Ich möchte in Verbindung mit meinem Keyboard Casio HT 3000 und der entsprechenden Software die notwendigen Grundlagen schaffen, selbst ein wenig zu komponieren und arrangieren. Ich habe zwar nicht vor, Dieter Bohlen zu ersetzen, aber eigene Musik zu kreieren, ist auch ganz schön. Können Sie mir bezüglich der zu beschaffenden Soft- oder auch Hardware weiterhelfen?

H.-G. Winkler, Markdorf

DTP ist eine faszinierende Möglichkeit, den Computer sinnvoll einzusetzen. Beim Amiga müssen dabei folgende Punkte beachtet werden: Professionelle DTP-Programme arbeiten aus Gründen der Übersichtlichkeit im hochauflösenden Modus (Interlace). Der Standard-Monitor ist für solche Arbeiten kaum geeignet. Inzwischen gibt es aber auch recht preiswerte Komplett-Pakete wie den Publisher (Test in der AMIGA DOS 03/90, Preis: zirka 300,– DM), die mit dem normalen Monitor arbeiten. Mit diesem Programm zum Beispiel lassen sich recht ansehnliche Ergebnisse erzielen. Bleibt das Problem des Druckers: Ein 24-Nadel-Matrix-Drucker gibt zwar zufriedenstellende, aber nicht optimale Ergebnisse ab. Hilfe würde das Programm PixelScript bringen (Test in diesem Heft), welches von Lay-

out-Programmen erzeugte PostScript-Dateien für Matrix-Drucker umwandelt und einen wesentlich besseren Ausdruck ermöglicht. Alles in allem kostet der Einsatz von DTP, wenn er zumindest semiprofessionell sein soll, nicht gerade wenig.

Die Lösung Ihres zweiten Problemes heißt MIDI (zu deutsch: Digitale Schnittstelle für Musikinstrumente). Falls Ihr Keyboard über eine MIDI-Schnittstelle verfügt, benötigen Sie ein MIDI-Interface, das den Datenaustausch zwischen Keyboard und Computer ermöglicht. Ferner benötigen Sie noch ein MIDI-fähiges Musikprogramm (beispielsweise Music-Studio von Activision oder das in der letzten Ausgabe vorgestellte Quest I). Ist diese Konstellation gegeben, können Sie Ihren Kompositionsfähigkeiten freien Lauf lassen. Im Musikbuch von DATA Becker ist übrigens eine Bastelanleitung für ein MIDI-Interface enthalten.

(Red.)

Wie rüste ich einen Amiga 1000 auf?

Wir benutzen einen Amiga 1000 und sind soweit mit Hard- und Software zufrieden. Das einzige Manko besteht in der langen Bootzeit und dem leider nicht ausreichenden Speicherplatz. In den meisten Fachzeitschriften wird immer wieder zu einer Festplatte geraten. Nun meine Frage:

Wie können wir dem 1000er Beine machen, ihn kostengünstig erweitern und welche Festplatte paßt zu ihm bzw. wo bekommt man sie?

B. Haidner, Wien

Die FSE- und CombiTec-Autoboot-Festplatten werden unter anderem auch für den Amiga 1000 angeboten. Die Kapazität der Platten liegt bei 20 bis 122 MByte.

Die Adressen:

CombiTec Computer GmbH
Liegnitzer Str. 6-6a
D-5810 Witten
Tel.:(02302) 88072

Frank Strauß Elektronik
Schmiedstr.11
D-6750 Kaiserslautern
Tel.:(0631) 67096-98

Stichwort "Pointer"

Ich möchte über Genlock einen Pfeil (wie auf der Amiga Workbench vorhanden) in ein laufendes Videobild einblenden. Leider ist im Amiga-Programm außer dem Pfeil noch eine Umrandung und eine Kopfleiste vorhanden. Kann man diese herauslöschen, bzw. nur einzelne Files von der Workbench kopieren, um dies zu erreichen, oder ist Ihnen ein anderes Programm bekannt, das nur einen Pfeil (Pointer) auf den Bildschirm bringt?

G. Held, Kutenholz

Falls der von Ihnen gewünschte Pfeil unbeweglich sein kann und nur an unterschiedlichen Stellen auftauchen soll, bietet sich die Möglichkeit an, den Pfeil mit einem Malprogramm zu erstellen und als IFF-Brush (Pinsel) abzuspeichern. Beim Einlesen des Genlock-Bildes können beide Screens aufeinander gelegt werden, wobei der Pfeil an die jeweilige Stelle gesetzt wird und alles zusammen auf das Videoband abgespeichert werden kann.
Soll es sich um einen beweglichen Zeiger handeln, muß ein Programm geschrieben werden, welches den neuen Pfeil anstelle des Mauspointers legt.

(Red.)

Amiga 500 und C64

Ich besitze einen Amiga 500 mit Speichererweiterung und gepufferter Uhr (Systemspeicher 1,5 MByte). Außerdem sind eine Festplatte (A590 20 MByte), ein Zweitlaufwerk (3,5 Zoll), ein Drittlaufwerk (5,25 Zoll) und ein Farbdrukker MPS 1500 angeschlossen. Durch Ihren Artikel "BASIC in allen Gassen" hat sich mir die Frage gestellt, ob es irgendwie möglich ist, die Software eines C64-Computers auf dem Amiga zu benutzen. Ich praktiziere dies bereits mit PC-Software (MS-DOS) mit Hilfe eines PC-Emulators (softwaremäßig) einigermaßen erfolgreich.
Gibt es einen C64-Emulator für den Amiga 500, können Sie mir Bezugsadressen nennen oder existieren hardwaremäßige Lösungen?

M. Kottsiepe, Kaarst-Vorst

Gleich zwei C64-Emulatoren waren noch vor gar nicht so langer Zeit auf dem Markt, einer von ihnen, der 'GO64', arbeitete mit den Programmen des C64 auch zufriedenstellend. Leider besitzen wir im Augenblick keine Informationen, ob diese Emulatoren noch auf dem Markt zu finden sind. Falls Ihnen, liebe Leser, noch etwas dazu einfällt, schreiben Sie uns bitte, wir werden die Infos in einer der nächsten Ausgaben veröffentlichen.

Betrifft: Programm GREGOR

Wenn ich die Erklärung zu Ihrem Programm GREGOR richtig verstanden habe, dann müßte das Jahr 1600 ein Schaltjahr sein. Nach der Eingabe von 29.2.1600 erhielt ich aber die Meldung "...gibt es nicht". Beim Suchen nach der Ursache glaube ich folgenden Fehler entdeckt zu haben.

Das Programm erklärt zwar in Zeile 15 jedes vierhundertste Jahr zum Schaltjahr, macht das aber in Zeile 22 wieder zunichte (jedes hundertste Jahr ist ein Nicht-Schaltjahr). Eine mögliche Abhilfe wäre die folgende Verlängerung der Zeile 22:

22: IF j/100 = INT(j/100) AND j400 < > INT(j/400) THEN

Bei mir funktioniert es jedenfalls.

A. Pistor, Wien

Probleme mit der DATABOX

Als Neuling, der im Besitz eines Amiga 500 ist, habe ich ein paar grundlegende Fragen zur DATABOX:

1) Mache ich von der DATABOX eine Kopie, verlangt diese nach dem Start immer das Original. Warum?

2) Wie starte ich BASIC ohne Workbench?

3) Wie kopiere ich ein Programm der DATABOX auf eine andere Diskette?

Peter Stiller
1000 Berlin 2

Höchstwahrscheinlich haben Sie Ihre DATABOX-Disketten durch Anklicken auf der Workbench kopiert. Dies ist schon richtig, jedoch müssen Sie die neuen Disketten mit den Original-Namen DATABOX-1 und DATABOX-2 benennen, das 'COPY OF ...' also entfernen. Die Programme werden von den logischen Laufwerken DATABOX-1: und DATABOX-2: gestartet, bei einer 'COPY OF...'-Version wird dieses logische Device nicht mehr erkannt.
Wie Sie BASIC ohne Workbench starten, lesen Sie in diesem Heft im Amiga-BASIC-Kurs unter der Überschrift 'BASIC-Diskette die Zweite'.
DATABOX-Programme lassen sich aufgrund der fehlenden Icons nur von der Shell

Die Problemecke

In der Problemecke werden Leserbriefe veröffentlicht, zu denen die Redaktion im Augenblick keine Antwort parat hat. Hier sind Sie, liebe Leser der AMIGA DOS gefragt. Sollten Sie zum einen oder anderen Thema eine Antwort wissen, schreiben Sie uns unter dem Stichwort 'Problemecke'.

Die PC-Karte im Blickpunkt

Die PC-Kartenbeschleuniger einiger Firmen sind mir schlichtweg zu teuer. In diesem Zusammenhang habe ich einige Fragen an Ihre Redaktion.

– Kann ich die Schaltung, die in der DOS 1'88, S. 48 erschienen ist, auf der PC-Karte 2088 zur Beschleunigung von 4,77 MHz auf 8 MHz verwenden?

– An welchen Custom-Chip und Pin muß ich die Schaltung anschließen?

– Welche Änderungen muß ich sonst an der PC-Karte vornehmen?

– Wo bekomme ich die neueste Version der JANUS-Software, die schon seit einem Jahr angekündigt wurde, her?

G. Bottke, Nürnberg

(CLI) aus kopieren. Dazu nehmen Sie den COPY-Befehl:
COPY DATABOX-1:<Programmname> TO RAM:
kopiert ein DATABOX-Programm von der Diskette ins RAM,
COPY ?
ruft den COPY-Befehl auf, ohne ihn auszuführen, danach eingeben:
RAM:<Programmname>
TO DF0:
kopiert auf eine danach eingelegte Diskette (sinnvoll, wenn nur ein Laufwerk vorhanden ist).
Ansonsten kann man mit COPY von jedem Laufwerk zu jedem Laufwerk kopieren.

(Red.)

Pascal-Ecke in der AMIGA DOS

Mit Interesse las ich Ihren Artikel in der Ausgabe 3/90 über KickPascal auf dem Amiga. Ich bin seit Ende Dezember '89 Besitzer dieses Compilers und trotz der von Ihnen aufgeführten Negativmerkmale sehr zufrieden. Das Ansprechen des Amiga-Betriebssystems ist gut gelungen und die Arbeitsumgebung sehr komfortabel. Zudem ist der Preis recht günstig. Deshalb würde mich eine Pascal-Ecke in Ihrer Zeitschrift besonders interessieren, und ich hoffe auf eine starke Resonanz seitens anderer Leser.

R. Ortner, Marl

Wir haben diesen Brief stellvertretend für alle diejenigen abgedruckt, die mehr über KickPascal in der AMIGA DOS wissen wollen. Unsere Aktion geht jedoch noch weiter: Schreiben Sie uns Ihre Meinung zu dieser 'neuen' Programmiersprache, wir sind weiter gespannt.

(Red.)

Datums- und Zeitangabe auf dem A500

Ich besitze seit einiger Zeit einen Amiga 500. Im Oktober letzten Jahres kaufte ich mir eine Speichererweiterung A 501 RAM von Commodore. Seither gab mein Amiga beim Starten der Workbench immer die aktuelle Datums- und Zeitangabe aus. Seit einigen Wochen nun lädt die Workbench zwar normal, aber anstatt der Datums- und Zeitansage erhalte ich den Ausdruck [unset] [unset] [unset],. Anfänglich dachte ich an eine Verseuchung durch Viren, aber eine Überprüfung mit der Original-Workbench brachte das gleiche Ergebnis. Könnte es eventuell sein, daß meine Speichererweiterung defekt ist? Die Startup-Sequence wurde von mir nicht geändert.

U. Mayer, Rothenburg o.T.

Ihre Speichererweiterung ist nicht defekt, das Problem liegt lediglich darin, daß Ihre batteriegepufferte Uhr neu gestellt werden muß. Stellen Sie dazu in den Preferences die Uhrzeit neu ein, und klicken Sie danach in diesem Menü den Save-Button. Dadurch wird die Zeit wieder auf den aktuellen Stand gebracht und gespeichert.

(Red.)

XT-Karte und deutscher Zeichensatz

Seit kurzem bin ich Besitzer eines Amiga 2000 mit PC-XT-Karte. Und genau dort liegt mein Problem. Nachdem ich MS-DOS geladen habe, ist leider nur die amerikanische Tastatur aktiv. Ich habe zwar gelesen, daß man in den Dateien CONFIG.SYS und AUTOEXEC.BAT entsprechende Änderungen vornehmen kann, aber damit habe hatte ich keinen richtigen Erfolg.
Deshalb möchte ich Sie bitten, mir doch den vollständigen Inhalt der beiden Dateien zu schreiben, um meinen Amiga auf deutsche Tastatur und einen Epsondrucker einzustellen. Nach dem Umschalten auf die deutsche Tastatur (im Direktmodus) konnte ich den Schrägstrich "\" auf keiner Taste mehr finden. Ist das ebenfalls korrekt?

F.Zawadski, Rastatt

Um den deutschen Zeichensatz auf der XT-Seite des Amiga zu aktivieren, sollten Sie die AUTOEXEC.BAT um die Zeile KEYB GR erweitern. Ferner tragen Sie dann in der CONFIG.SYS für COUNTRY die Zahl 049 ein. Den Backslash (\) erreichen Sie übrigens mit der Taste über dem Tabulator (links neben der 1). Der Drucker dürfte nun auch keine Schwierigkeiten mehr machen.

(Red.)

Installationsschwierigkeiten

Vor kurzem legte ich mir die PD-Disketten Fish-Disk Nr. 67 und 158 zu. Als Anfänger auf dem Amiga stellte sich die Installation der Programme DiskX (Fish 158) und Wheel (Fish 67) für mich als sehr schwer heraus. Nachdem ich zum Beispiel DiskX geladen hatte, erscheint im CLI, daß diese Version ARP V33 benötigt.
Unter dem Icon DiskX.Note steht: "Copy AmigaLibDisk 158:Libs/ARP.Library Libs:". Nachdem ich dieses durchführen wollte, meldete mir das CLI, daß die Datei nicht vorhanden ist.
Bei dem Programm Wheel verstehe ich bei der Installation nur Bahnhof...

T. Rubbert, Hamburg

Bei dem Programm DiskX auf der Fish 158 handelt es sich um die Version 2.1, die die ARP-Library benötigt. Bei dieser Library handelt es sich um einen CLI-Ersatz, der sich beispielsweise auf den PD-Disketten Panorama 27c und RPD 122 befindet. Besorgen Sie sich diese ARP-Library, und kopieren Sie sie in das Libs:-Verzeichnis der Fish-Disk 158. DiskX sollte dann ohne Probleme laufen.
Bei dem Programm Wheels handelt es sich um ein Spiel, das auf den AmigaBasic-Interpreter zurückgreift. Zu diesem Behufe sollten sie das AmigaBasic auf die Fish-Disk 67 kopieren. Auch dieses Programm sollte nun keinerlei Probleme mehr darstellen.

(Red.)

Fehler im Etiketten-Printer

Ich mache mir hin und wieder die Mühe und tippe ein interessant erscheinendes Listing ab, wenn es nicht zu lang ist. Und das, obwohl ich der Sprache BASIC kaum mächtig bin. Hier fiel mir aber auf, daß das Programm nur laden und speichern kann, wenn im gleichen Verzeichnis ein Directory namens 'DISKS' auftaucht, welches man sich selbst erstellen muß (zum Beispiel 'MAKEDIR SYS:DISKS'). Die Druckroutine entspricht nicht meinen Erwartungen an ein Normaletikett für 3,5-Zoll-Disketten im Format 70 x 70 mm. Es wäre doch schön, wenn in den ersten beiden Zeilen für die Rückseite der Disk folgender Text eingebracht werden könnte:
Enable - >
Protect - >
Offensichtlich hat der Autor Etiketten im Format 50 x 70 mm (oder ähnliches) - also nur für die Vorderseite - verwendet. Dementsprechend habe ich mit meinem EPSON-kompatiblen Drucker Schwierigkeiten mit überflüssigen Linefeeds. Eine Bitte daher: Gebt doch mit dem Hinweis auf das anzulegende Verzeichnis auch die Anregung weiter, die Druckroutine so umzuschreiben, daß auch das von mir benutzte Format unterstützt wird.
Ich würde mich freuen, in einer der nächsten Ausgaben etwas in dieser Richtung zu erfahren.

Wilhelm Ditsche
Buchholz 2

Beim Etiketten-Printer kam es tatsächlich zu einem Abbruch des Programms, wenn das Directory 'DISKS' nicht gefunden wurde. Dieses Verzeichnis steht in Zeile 490 und kann nach Belieben verändert werden. Im Text wurde dies jedoch leider nicht erwähnt.
Zum Ausdruck ist zu sagen, daß uns hier nur eine begrenzte Anzahl Drucker zum Austesten zur Verfügung steht. Der eine oder andere Leser wohl deshalb irgendwann einmal Schwierigkeiten bekommen, wenn sein Drucker es mit der EPSON-Kompatibilität nicht so genau nimmt. In dem Fall bleibt es dem geplagten Druckerbesitzer nicht erspart, sich an die Arbeit zu begeben und das Drucker-Handbuch zu wälzen.
Leider hat uns Herr Ditsche nicht mitgeteilt, welchen Drucker er besitzt, in einem solchen Fall wäre es sinnvoll, Drucker- Probleme in die Problemecke zu setzen. Sollte jemand von Ihnen, liebe Leser(innen), das ETTI-Programm so verändert haben, daß es wie von Herrn Ditsche gewünscht funktioniert, so würden wir uns freuen, wenn Sie uns schreiben.

(Red.)

★ Bitte beachten Sie folgendes:

Bitte schreiben Sie bei Leserbriefen Ihre vollständige Adresse nicht nur auf den Umschlag, sondern auch auf den Brief; nur so ist eine schnelle Beantwortung möglich. Bei Briefen, die in der Leserbrief-Rubrik abgedruckt werden, behält sich die Redaktion vor, Leserzuschriften in gekürzter Form wiederzugeben.

(Red.)

Ein toller Dreh

Kubistisches

Von allen Faszinationen gehört die Faszination der Dimensionen sicherlich zu den spannendsten. Wir, die wir glauben, ein Leben in vier Dimensionen zu verbringen (nämlich in den drei Dimensionen des Raumes und in der der Zeit), sind beispielsweise nicht in der Lage, uns in mehr- oder minderzahlige Dimensionen hineinzudenken.

Stellen Sie sich beispielsweise einmal ein Wesen vor, das in der fünften Dimension (was immer das auch sein mag) existiert oder eines, das mit zweien auskommt. Bis auf ein paar begnadete Genies, die behaupten, es zu begreifen, wird die Mehrzahl doch an dieser Vorstellung scheitern. Das ist wenig verwunderlich, da die Zahl derjenigen, die Probleme mit der räumlichen Vorstellung haben, gar nicht einmal so gering ist.

Wir wollen uns in dieser Tüftelecke einmal in die drei Dimensionen eines Würfels begeben und unsere Vorstellungskraft zum Rotieren bringen.

Die Aufgabe

Wir stellen uns einen Würfel vor, der seinerseits aus 343 kleineren Würfeln besteht (also je sieben Würfel lang, breit und hoch ist).

Um eine bessere Orientierung innerhalb dieses Gebäudes zu bekommen, geben wir den einzelnen Würfeln Kennungen in Form von drei Zahlen, die die Koordinaten der kleineren Würfel im Raum darstellen. Der Würfel in der unteren linken vorderen Ecke wird demnach 1-1-1 heißen, der in der oberen rechten hinteren Ecke 7-7-7. Sollte sich im Verlauf unserer Reise der Würfel zu drehen beginnen, dann erhält der neue untere linke vordere Teilwürfel die Kennung 1-1-1; die anderen Teile werden entsprechend neu benannt.

Wir betreten nun den Würfel mit der Nummer 1-1-1. In der unteren Box wird ein Weg beschrieben, den wir durch das Innere des geometrischen Aufbaus schreiten wollen, wobei der Würfel die schlechte Eigenschaft hat, sich derweilen um eine seiner Achsen drehen zu müssen (siehe Zeichnung).

Unsere Frage:
– In welchem Würfel befinden wir uns am Ende des Weges?

Die Lösung der Frage schreiben Sie bitte auf eine Postkarte und schicken diese an den:

DMV-Verlag
Redaktion AMIGA DOS
Stichwort "Kubus"
Postfach 250
3440 Eschwege

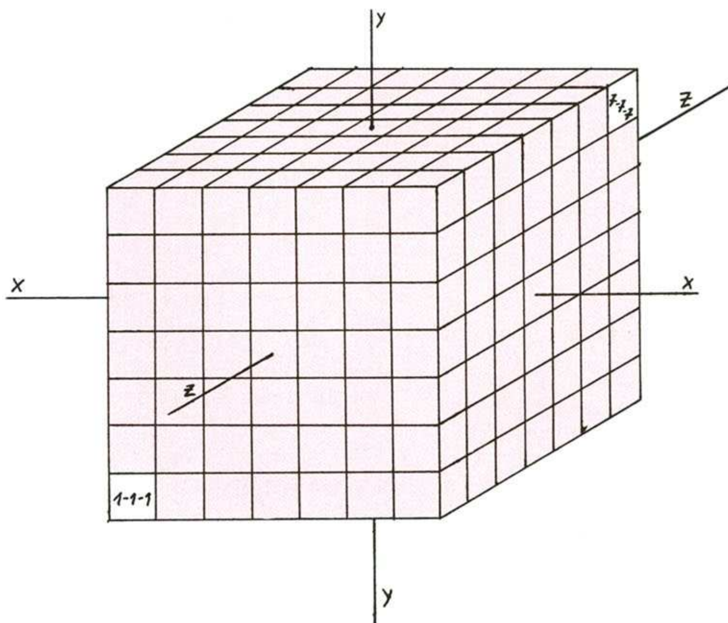
Der Rechtsweg ist ausgeschlossen. Mitarbeiter des DMV-Verlages und deren Angehörige dürfen leider nicht mitmachen. Der Einsendeschluß ist der

15. Mai 1990.

Es gilt das Datum des Poststempels.

Was gibt es zu gewinnen?

Unter allen richtigen Einsendungen verlosen wir diesmal 10 Exemplare des Spiels X-Out von Rainbow Arts.





Der Weg:

– voran:	3 Schritte
– rechts:	4 Schritte
– hoch:	2 Schritte
– zurück:	2 Schritte

Ein Ruck geht durch den Würfel, und dieser dreht sich an der z-Achse um 90 Grad links herum.

– voran:	1 Schritt
– links:	3 Schritte
– hinab:	3 Schritte

Und wieder ruckt es im Gebälk: Der Würfel dreht sich an der x-Achse 90 Grad nach hinten.

– hinab:	4 Schritte
– rechts:	3 Schritte

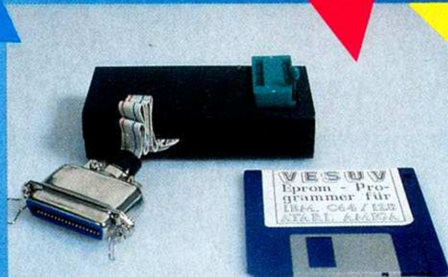
Ein letztes Mal erzittert das Gebilde, und der Würfel dreht sich erneut an der z-Achse um 90 Grad nach links. Danach verharrt alles in der Stille, und ein letztes Echo der Rotation verhallt in den Tiefen des Würfels.
Wo aber befinden wir uns jetzt? (jf)



VIRUS-FALLE 29,95 DM
verhindert das Ausbreiten von Boot-Viren.

LIGHTPEN ohne Maustasten **79 DM**

KICKSTART 3 **59,95 DM**
Umschaltplatine für 3 verschiedene Kickstarts ⊕ 2x original Kickstart-Roms und 1x in Eproms ⊕ Roms/Eproms nicht im Lieferumfang enthalten

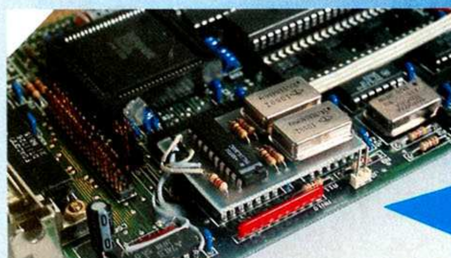


VESUV-AMIGA-Eprommer 199 DM
läuft auf A500, 1000 und 2000 ⊕ brennt auch 1 MBit-Eproms ⊕ „HAPPY" 3/89 Test-Gesamturteil „SEHR GUT" ⊕ programmiert die Eproms 2716-27512, 27513 und 27011



MEGA-DRIVE **299 DM**
2 MByte Diskettenlaufwerk für AMIGA-Dos! ⊕ 1,52 MByte unter Amiga-DOS ⊕ arbeitet auch mit Ihren alten 880 k Disketten ⊕ abschaltbar ⊕ durchgeschleifter Bus

10 HD-Disketten (1,4 MB) **29,95 DM**



TURBO-[illegible] (ca. 50 % schneller) **398 DM**
TURBO-XT (ca. dopp. so schnell) **199 DM**
XT-RAM 256 k **298 DM**
⊕ erweitert Ihre XT-Karte ON BOARD auf 768 KByte! ⊕ AT/XT-Karte nicht im Lieferumfang

Power PC-Board **798 DM**
MACHT IHREN AMIGA 500 IBM-KOMPATIBEL ⊕ echte 16-Bit-CPU V 30 bei einer Taktfrequenz von 8 MHz ⊕ Phoenix-Bios mit 768 k Ram unter MS-DOS (im Lieferumfang enthalten) ⊕ alle Amiga Ein- und Ausgänge werden unterstützt: Maus, Joystick, interne und externe (auch 5,25 Zoll) Laufwerke, parall./seri. Schnittstelle ⊕ Festplatten ⊕ Superschnelle Bildschirmausgabe: unterstützt Herkules und Farbgrafik! ⊕ 1 MByte Ram und Uhr ON BOARD; auch für den Amiga 500 ansprechbar ⊕ komplett mit MS-DOS 4,01; GW-BASIC; SHELL; 1MB Ram; Uhr; dt. Handbücher

AMIGA 500



A 512 **179 DM**
512 k Speichererweit. ⊕ abschaltbar ⊕ Uhr

A2MB/500 **598 DM**
2 MByte Ramkarte ⊕ mit FAT-AGNUS 1,8 MByte mit BIG-AGNUS volle 2 MByte (Chipram/Fastram) ⊕ WELTNEUHEIT: arbeitet mit dem BIG- und dem FAT-AGNUS!



AMIGA 1000



A8MB/1000 **798 DM**
8MByte Ramkarte; mit 2MByte bestückt ⊕ einfachster Einbau ⊕ kein Löten - nur einstecken ⊕ abschaltbar ⊕ ohne Waitstates

AMIGA 2000



A8MB/2000 **698 DM**
8 MByte Ramkarte mit 2 MByte bestückt ⊕ zukunftssicher durch 4-MBit-Technologie ⊕ autokonfigurierend ⊕ 0-Waitstates ⊕ abschaltbar ⊕ Anschluß für Reset-Taster

Multiboard Ramkarte 2MB **898 DM**
mit 4 MByte bestückt **1398 DM**
mit 8 MByte bestückt **1998 DM**
incl. 4-fach Kickstart-Umschaltung!



...bei uns nutzen Sie heute Technologie von morgen

Bitte fordern Sie unseren Gratiskatalog an!

BIG AGNUS
mit Einbauanleitung für A 500/2000
BEI UNS NUR
99 DM!



HOTLINE

Technische Fragestunde: Mo.-Fr. von 16-17 Uhr. Hier können Sie die Entwickler unserer Amiga-Produkte sprechen.

0 22 25/20 61-20 62-20 63

Neuer Markt 21 5309 Meckenheim Telefon 0 22 25/20 61-6



Mit Deluxe Paint wie ein Profi arbeiten

Durch Menüs haben wir uns gekämpft, in Icons geschwelgt, Requester aufgerufen und Parameter gesetzt. Wir haben in den bisherigen vier Werkstattbeiträgen das elementare Arbeiten mit Deluxe Paint und seinen unterschiedlichen Funktionen kennengelernt. Die letzte Routine erlangen Sie jedoch nicht durch endloses Studium des Handbuches, sondern durch angewandte Praxis.

Nach längerem Arbeiten mit Deluxe Paint erarbeitet der Anwender sich viele kleine Tricks, die den Umgang mit Grafik und Deluxe Paint erleichtern.

Kursfahrplan

Teil 1: Das Basismenü und seine Funktionen

Teil 2: Die Kopfleiste und die dort enthaltenen Menüs

Teil 3: Die Anwendung der in der Kopfleiste enthaltenen Funktionen

Teil 4: Animation mit dPaint III

Teil 5: Arbeitstips vom Profi

In diesem letzten Teil wollen wir uns eben diesen kleinen Praxistips zuwenden. Nicht in jedem Falle müssen diese Tips unmittelbar die Arbeit mit Deluxe Paint betreffen. So leistungsfähig das Programm auch ist, in manchen Bereichen, die den Computergrafiker brennend interessieren, muß auch Deluxe Paint passen. Darum werden wir in diesem Teil der Werkstatt nicht Deluxe Paint an sich, sondern noch andere Software ansprechen, deren Leistungen für Deluxe-Paint-Benutzer interessant sind. Wer viel mit Deluxe Paint arbeitet, wird mit der Zeit feststellen, daß die Bedienung des Programmes über die Menüleiste zwar recht komfortabel ist, richtiges, schnelles Arbeiten damit aber etwas problematisch wird.

Rationelles, schnelles Arbeiten

Werden die gewünschten Kommandos über die Tastatur aufgerufen (fast jeder Deluxe-Paint-Befehl hat ein Tastaturäquivalent), stehen die gewünschten Funktionen fast augenblicklich zur Verfügung.

Natürlich läßt sich nun einwenden, daß hier doch einiges an Tastaturcodes auswendig zu lernen ist, bis Deluxe Paint komplett von hier aus bedient werden kann. Allerdings sollten Sie bedenken, daß sich aus der Fülle der Deluxe-Paint-Funktionen einige herauskristallisieren, die viel öfter benutzt werden als andere. Auch wenn nur diese wenigen verstärkt benötigten Befehle über die Tastatur aktiviert, die wesentlichen anderen Optionen aber wie bisher



Deluxe Paint und Butcher parallel im Einsatz

durch die Menüs bedient werden, ist schon eine deutliche Zeitersparnis zu verzeichnen.

Sie sollten auf jeden Fall versuchen, sich ein Blatt mit den einzelnen Tastaturcodes neben den Computer an die Wand zu hängen und jedesmal, wenn eine neue Deluxe-Paint-Funktion aktiviert wird, nachsehen, welche Taste dafür steht. Auf diese Weise lernen Sie diese Codes recht schnell, und ehe Sie sich versehen, kennen Sie die meisten und können Deluxe Paint weit komfortabler als bisher bedienen.

Dateien hin, Dateien her

Wenn länger an einem besonders aufwendigen Motiv gearbeitet wird, fallen im Laufe der Arbeit immer mehr Bild- und Brushdateien an, die angelegt wurden, um Zwischenschritte zu sichern. Schnell hat man auf diese Art und Weise ein Verzeichnis gut gefüllt. Deluxe Paint lädt jedesmal, wenn ein <Load>-Requester geöffnet wird, alle Directory-Einträge des Verzeichnisses ein, auf das der Requester gerade eingestellt ist. Sind hier nun viele lange Einträge, resultieren daraus lange Ladezeiten. Dem kann vorgebeugt werden, wenn zum einen versucht wird, den Dateinamen so kurz wie möglich zu halten, zum anderen mit mehreren verschiedenen Verzeichnissen gearbeitet wird, und so die Länge der Directorys beschränkt wird.

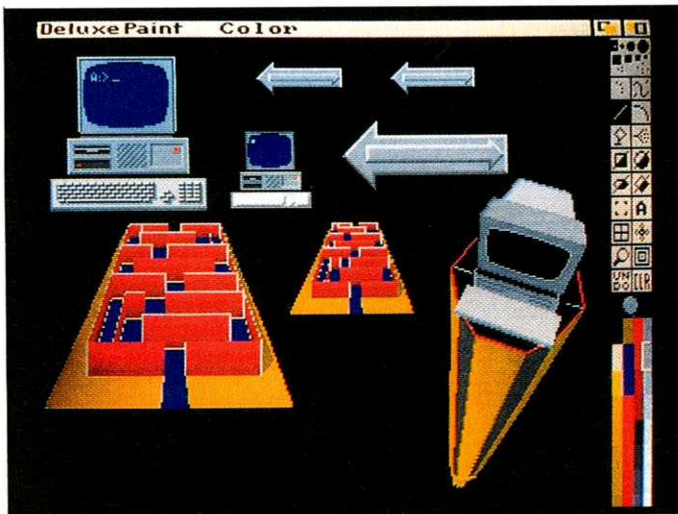
Natürlich bezieht sich dies im wesentlichen auf die Arbeit mit Diskettenlaufwerken, Festplatten haben in der Regel eine genügend kurze Zugriffszeit auf die Daten, so daß Lesevorgänge auch bei wirklich umfangreichen Directorys in einem zeitlich vertretbaren Rahmen bleiben.

Noch eine Warnung vor zu langen Dateinamen und Leerzeichen in Dateinamen. Tatsächlich ist es schon passiert, daß Verzeichnisse, die randvoll mit überlangen Dateinamen waren, sich aus unerfindlichen Gründen nicht mehr laden ließen. Benutzen Sie Leerzeichen in Dateinamen und wollen diese Dateien später mit der Shell oder dem CLI bearbeiten, müssen Sie darauf gefaßt sein, daß sich auch hier vermeidbare Probleme einstellen. Auch die Benutzung der RAM-Disk ist empfehlenswert, wenn es um schnelles Zwischenspeichern geht. Die RAM-Disk braucht etwa 1.5 Sekunden um ein 32-Farben-Bild im HighRes-Modus einzuladen. Allerdings sollten Sie nicht vergessen, die RAM-Disk nach wichtigen Bildern zu durchsuchen, bevor Sie einen Reset ausführen oder den Computer abschalten.

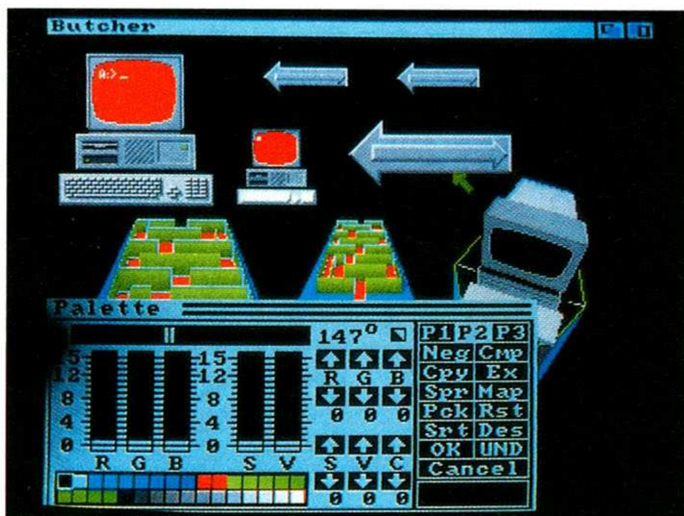
Das Bildarchiv, des Grafikers Bibliothek

Wird der Computer des öfteren zum Malen benutzt, fallen mit der Zeit eine Vielzahl von Bildern an, ganz zu schweigen von den Brushes, die als Zwischenschritte während der Ausarbeitung des fertigen Bildes anfallen. All diese Grafiken können nun entweder auf Diskette vergammeln oder aber zu einem Archiv zusammengestellt werden, das bei späteren Abeiten herangezogen werden kann. Ein solches Archiv erfordert natürlich ein gewisses Maß an Organisation. Am besten ist es, nach Abschluß jedes Projektes die vorhandenen Motive zu sichten. Kleinere Brushes können auf einem Bildschirm zusammengestellt und abgespeichert werden. Dies geht natürlich nur, wenn alle Brushes mit denselben Farben arbeiten. Wird eine Brush aus einer anderen Palette hier hineingesetzt, übernimmt sie natürlich die aktuell eingestellten Farbwerte und sieht dann entsprechend poppig bunt aus. Beginnt das Archiv zu wachsen, stellt sich bald die Frage nach der Organisation. Die Frage “Wie finde ich in meinem Archiv etwas?“ gewinnt an Bedeutung.

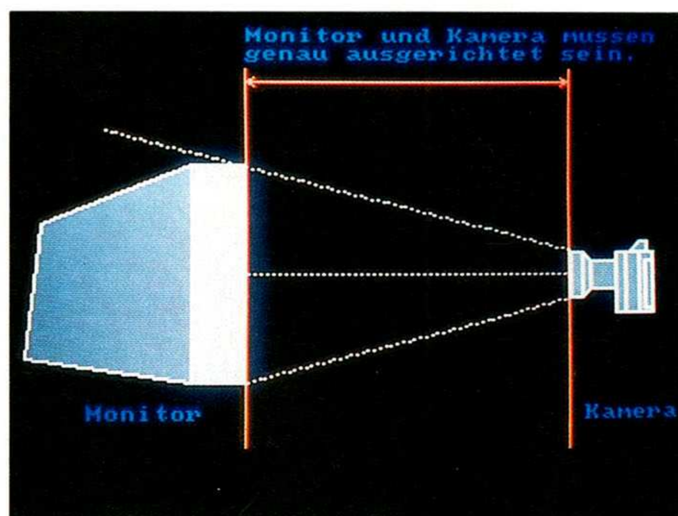
Sollten Sie einen Drucker besitzen, ist es eine gute Idee, jeden kompletten Bildschirm,



Solcherart können auf einem Archivbild viele verschiedene Brushes platzsparend zusammengestellt werden



Kontrolle über Farben und Register – Butcher ist die ideale Deluxe-Paint-Ergänzung



Ausrichtung einer Kamera zum Monitor bei Bildschirmfotos

den Sie mit Bildteilen gefüllt haben, einmal auszudrucken, den genauen Dateinamen und den Namen der Diskette darauf zu notieren und die Ausdrucke zu sammeln.

Wird dieses Konzept konsequent verfolgt, können die Drucke in einem Ordner abgelegt werden, und fortan suchen Sie Ihre Bilder nicht mehr auf Diskette, sondern in Ihrem Archivordner.

Festplatten zeichnen sich zwar durch einen schnellen Datenzugriff aus, von daher ist das Durchsuchen diverser Dateien nach einem bestimmten Motiv weniger zeitraubend als mit einer Diskettenstation. Trotzdem empfiehlt sich auch hier ein Archiv, und sei es nur aus Gründen der Übersichtlichkeit.

Interessantes über IFF

Als sich die Programmierer von Electronic Arts bei der Arbeit an Deluxe Paint der Problematik widmeten, Grafiken auf Diskette zu speichern, stellte sich die Frage, in welchem Format dies geschehen soll. Die Lösung war das IFF-Format. Hier wurden einige wichtige Vorgaben erfüllt, die eine weitere Verarbeitung der Bilder ermöglichen.

IFF – das steht für Interchange File Format, die Dateistruktur eines IFF-Bildes wird von fast jedem anderen Programm erkannt. Eine IFF-Datei kann aber nicht nur Grafikdaten enthalten. Das IFF-Format ist durchaus auch in der Lage, Texte, Sounddaten und Soundsamples abzuspeichern. Rein theoretisch kann eine IFF-Datei sogar gleichzeitig Grafikdaten, Sounddaten und Textdaten enthalten.

Butcher – Der Grafikmetzger

Schon bald wird der Computergrafiker feststellen, daß Deluxe Paint ihm zwar die Kontrolle über die Farbe verleiht, ein weitergehender Zugriff auf die Farbregister allerdings verwehrt bleibt. Insbesondere in diesem Bereich eröffnet das Programm Butcher von Eagle Tree Software völlig neue Arbeitsperspektiven.

Stellen Sie sich bei der Arbeit mit Deluxe Paint folgenden Problemfall vor. Sie haben mit der Spritzpistole gearbeitet und auf einem fast fertigen Bild eine Weile zwei verschiedene Farben versprüht. Plötzlich stellen Sie fest, daß die beiden zuletzt benutzten Zeichenfarben eigentlich hätten vertauscht werden müssen. Ginge es nun um zwei große Flächen, könnten wir das Problem von Deluxe Paint aus mit zwei Füllaktionen beheben. Aber hier geht es um viele hundert verschiedene Pixel, die keine zusammenhängende Fläche ausmachen. Die einzige Methode, das Problem schnell zu lösen, wäre, einfach die Farbwerte in der Palette neu zu regeln. Wurde die Farbe aber außerdem noch in anderen Motivteilen verwendet, verändern wir deren Farbwerte natürlich auch, also wieder Fehlanzeige mit der schnellen Lösung.

Wirklich fix behoben ist der Sorgenfall mit dem Butcher. Er versetzt den Anwender in die Lage, Farbregister zu vertauschen, womit unser Problem behoben wäre. Aber der Butcher hat noch andere Vorzüge. Ist ausreichend Speicher vorhanden, können Butcher und Deluxe Paint gleichzeitig betrieben werden. Beide Programme unterstützen, falls vorhanden, RAM-Disks, so daß sich der Datentransfer zwischen den beiden Tasks zeitsparend über dieses Speichermedium abwickeln läßt. Ohne RAM-Disk müssen die Bilder auf Diskette gespeichert, und vom anderen Programm wiederum von dort gelesen werden.

Der Butcher kann Bitplanes und Farben separieren, er arbeitet mit drei frei wählbaren, unterschiedlichen Paletten gleichzeitig, er sortiert die Pixel eines Bildes nach Farbe und Menge und kann außerdem noch vieles mehr. Tatsächlich hat sich der Butcher 2.0 in der Praxis als ideale Ergänzung zu Deluxe Paint herauskristallisiert. In diesem Zusammenhang sei er auch anderen Deluxe-Paint-Benutzern ans Herz gelegt.



Der AMIGA-DOS-Animator ermöglicht Animationen auch mit Deluxe Paint II

Deluxe Paint und Video

Eines der beliebtesten Peripheriegeräte, das auf den Wunschzetteln von Computergrafikern ganz oben steht, ist ein Videodigitizer. Scheinbar das ideale Bilderfassungsgerät, um Vorlagen vom Papier in den Computer zu übertragen.

Den Grafiker wird dabei wohl weniger am eigentlichen Vorgang des Digitalisierens interessiert sein. Sein Thema wird die nachträgliche Modifikation der Bilder sein. Dabei stellt sich gleich zu Anfang ein elementares Problem. Wirklich gute Ergebnisse

werden nur erzielt, wenn mit möglichst vielen Farben digitalisiert wird. Werden weniger Farben gewählt, sollte das Motiv einen möglichst guten Kontrast haben. Andererseits kann aber selbst Deluxe Paint III (bei Deluxe Paint II nicht vorhanden) nur maximal im <Halfbright>-Mode arbeiten, und der stellt, wie Sie sich sicher erinnern können, 64 Farben dar.

Um die Bilder mit Deluxe Paint zu verarbeiten, muß beim Digitalisierungsvorgang die Farbenanzahl auf 32 bzw. 64 eingestellt werden.

Einige Digitalisierungs-Softwarepakete legen die Digigrafik nicht im IFF-Format ab, so daß das Bild nicht ohne weiteres in Deluxe Paint eingelesen werden kann. Hier schafft das Programm Butcher Abhilfe. Als eines der besten Werkzeuge, deren Leistungsumfang über das von Deluxe Paint hinausgeht, werden wir uns diesem Programm später noch ausführlich widmen. An dieser Stelle interessiert uns jedoch nur die Fähigkeit des Butchers, die unterschiedlichsten Grafikdateiformate einlesen zu können. Im Programm kann dann der Bildschirmmodus geändert werden, und das Motiv wird auf das neue Format und dessen Farbanzahl konvertiert. Die Grafik kann nach den Veränderungen als IFF-Datei abgespeichert werden, die ohne Probleme in Deluxe Paint eingeladen werden kann.

Video und Animation

Wer dem letzten Teil unserer Werkstatt aufmerksam gefolgt ist, wird festgestellt haben, das insbesondere die Erstellung der vielen verschiedenen Bilder sehr zeitaufwendig ist. Das Digitalisieren von Bildvorlagen kann auch hier Zeit sparen helfen. Natürlich können Sie kein Foto mittels eines Digitizers und einer Videokamera in Bewegung versetzen. Aber es gibt andere Möglichkeiten, die hier genutzt werden können. Ein Beispiel: Ihre Aufgabe ist es, einen Theatervorhang zu zeichnen, der nach den Seiten gerafft wird. Sie sollen nun eine Animationssequenz erstellen, die das Raffen der Vorhänge darstellt. Nun könnte man darangehen, per Manus einen Vorhang zu zeichnen und die einzelnen Phasen Stück für Stück zu erstellen. Ein Arbeitsaufwand, der einen versierten Computergrafiker gut eine Woche lang beschäftigen kann. Allerdings hätte dieser Grafiker auch Videokamera und Digitizer bemühen können. Denn, nimmt man ein Stück Stoff und arrangiert es vor der Kamera so, daß in etwa der Anschein eines Theatervorhangs erweckt wird, können auf diese Art und Weise die einzelnen Grundphasen innerhalb von wenigen Stunden als Computergrafik vorliegen, die natürlich noch manuell bearbeitet werden müßten. Aber selbst wenn man diesen Zeitaufwand berücksichtigt, ist die Erstellung der Animation unter Zuhilfenahme der Videokamera weit effektiver. Solche Bilder können mit Comic Setter realisiert werden.

Gibt es neben der vernünftigen Software und der entsprechend leistungsfähigen Hardware noch weitere Dinge, mit welchen sich unsere Amiga-Grafik-Workstation sinnvoll ergänzen ließe?

Immerhin gibt es da noch Lichtstifte, Trackballs und Grafiktablets. Lichtstifte haben sich in der Praxis bisher nicht bewähren können und Trackballs können, was die Präzision beim Positionieren des Cursors anbetrifft, nicht mit einer normalen Maus mithalten.

Extrahardware für Deluxe Paint?

Grafiktablets sind ebenfalls recht ungenau und obendrein auch noch fast immer inkompatibel zu Deluxe Paint.

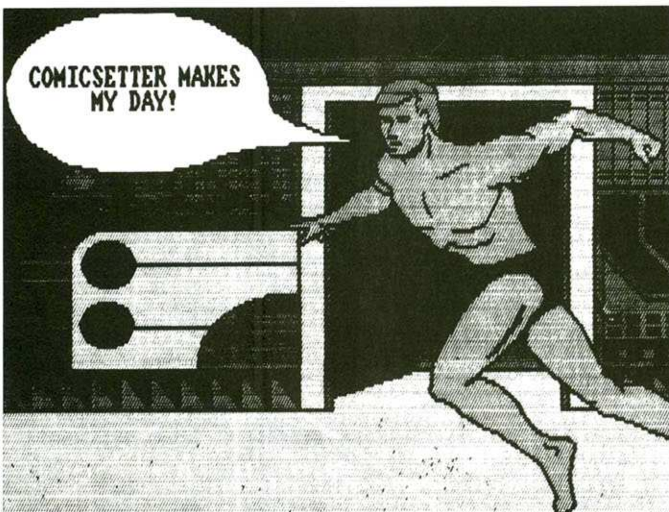
Tatsächlich hat sich in der Praxis die Maus ausgezeichnet bewährt. Wer an dieser Stelle noch Investitionen tätigen möchte, sollte sich also besser eine funktionelle Maus zulegen.

Das zukünftige Entwicklungen uns noch Eingabemedien bescheren werden, die das arbeiten mit Deluxe Paint noch komfortabler gestalten, ist nicht ausgeschlossen. Doch bis zur Stunde kann, außer der Maus, kein anderes wirklich überzeugen.

Fotos vom Bildschirm

Kein Drucker kann ein computergrafisch erstelltes Bild in einer Qualität wiedergeben, die mit einem guten Bildschirmfoto konkurrieren könnte. Doch allein die Bildschirmfotografie ist alles andere als leicht. Wer von dieser Methode Gebrauch machen möchte, muß zunächst einmal über eine gewisse Ausrüstung verfügen. Dazu gehören neben einer Spiegelreflexkamera ein Stativ, und wenn möglich ein Fernauslöser.

Die besten Ergebnisse werden erzielt, wenn die Kamera zirka 1,5 Meter vom Monitor aufgestellt wird. Achten Sie darauf, daß die Fläche des Bildschirms im Verhältnis zum Kameraobjektiv möglichst parallel ausgerichtet ist, andernfalls kommt es auf dem Foto zu unschönen Verzeichnungen und Deformationen des Motives. Während der Aufnahme sollten möglichst alle anderen Lichtquellen im Raum abgeschaltet werden. Sollten Sie keine Rollos oder vergleichbares haben, ist es im Interesse eines lohnenden



Der Comic Setter, ein Grafikprogramm dessen Leistungen sich von Deluxe Paint unterscheiden

Bildergebnisses sogar ratsam zu warten, bis es draußen dunkel geworden ist. Eine Eigenart der Bildröhre ist der große Anteil blauen Lichtes, den der Kathodenstrahl der Röhre erzeugt. So ist auch, wenn die Bildröhre nur einen weißen Bildschirm zeigt, immer ein sehr starker Blauanteil vorhanden. Ein Kunstlichtfilm mit entsprechenden Voraussetzungen kann Abhilfe schaffen. Zusätzlich ist auch der Helligkeitswert des Bildes ein Faktor, der berücksichtigt werden muß. In der Praxis hat sich herausgestellt, daß es ein Fehler ist, knauserig zu sein. Statt einem Foto pro Motiv sollten Sie mehrere mit verschiedenen Belichtungszeiten anfertigen. Zum einen ist dann sicher ein Bild gut geworden, zum anderen fallen viele Vergleichswerte an, anhand derer spätere Fotosessions leichter von der Hand gehen, und wahrscheinlich noch bessere Resultate zeitigen.

Jenseits von Deluxe Paint

So leistungsstark und komfortabel Deluxe Paint ist, viele Grafikmöglichkeiten des Amigas bleiben diesem Programm verschlossen.

Wer mit dem Hold and Modify Mode oder dem Interlace Mode arbeiten will, muß auf andere Software zurückgreifen.

Während Deluxe Paint ein Bitmap-orientiertes Grafikprogramm ist, also in bestimmten Grenzfällen für die Darstellung eines Pixels fünf Bitmaps benötigt werden, gibt es auch objektorientierte Grafikprogramme. Hier wird jeder Zeichenvorgang nur auf einer Bitmap dargestellt und ist dementsprechend einfarbig. Da hier auch die Bildschirmauflösung weit höher gewählt werden kann (das Flimmern bleibt allerdings auch bei diesen Programmen und kann nur durch einen Fickerfixer unterdrückt werden), eignen sich diese Programme sehr gut zum Erstellen von Strichgrafiken, die dann im Rahmen von Desktop Publishing weitere Verwendung finden. Ein empfehlenswertes Programm dieser Art ist Professional Draw von Gold Disk.

Ein anderes recht eigenwilliges 'Grafikprogramm' ist der Comic Setter. Dieses Programm ist ausschließlich auf das Erstellen von Computercomics ausgelegt.

Neben der rein grafischen Seite bietet Deluxe Paint auch einige Animationsfeatures. Natürlich gibt es auch eine ganze Reihe von Programmen, die ausschließlich der Animation dienen. Eine sinnvolle Ergänzug von Deluxe Paint ist Zoetrope von GFA Systemtechnik.

Animationen mit Deluxe Paint II?

Leider beinhaltet Deluxe Paint II nicht die guten Animationsfeatures der IIIer Version. Aber es gibt eine Möglichkeit, die auch den Deluxe-Paint-II-Anwendern Animationen ermöglichen.

In der Ausgabe 4/90 der AMIGA DOS finden Sie ein Assemblerlisting, das einen Animator als zweiten Task neben Deluxe Paint eröffnet. Sie können diesen Animator während der Arbeit mit Deluxe Paint aufrufen und auf dem Deluxe-Paint-Zeichenbildschirm enthaltene Animationen ablaufen lassen.

Finale

Unsere Deluxe-Paint-Werkstatt wurde zu einem ausgedehnten Streifzug durch die Wunderwelt der Amiga-Grafik. Sie haben gesehen, wie mit diesem Programm die tollsten Effekte gezaubert und die fantastischsten Bilder erstellt werden können. Jetzt sind Sie gefordert; Ihre Kreativität und Ihre Phantasie.

Auch wenn die Deluxe-Paint-Werkstatt mit diesem Teil schließt, werden wir uns diesem weiten Gebiet der Computeranimation in anderem Rahmen wieder nähern. Bis dahin wünschen wir Ihnen viel Spaß mit Deluxe Paint und Ihren Bildideen.

Im Rahmen unseres Pixel-Panoramas, das Sie auch in dieser AMIGA DOS finden, haben wir einen Grafikwettbewerb gestartet, an dem Sie sich mit Ihren Bildern beteiligen können. Wir freuen uns schon auf Ihren Beitrag.

(hs)



Übersicht über die wichtigsten Deluxe Paint Tastaturkommandos

Das Basismenü

b	Zeichenstifte definieren
B	- - - -
c	Ungefüllte Kreise
C	Gefüllte Kreise
d	Kontinuierliches Freihandzeichnen
D	Gefülltes Freihandzeichnen
e	Ungefüllte Ellipse
E	Gefüllte Ellipse
f	Füllen
F	Füllart Requester
g	Gitter an/aus
G	Gitter an/aus, wobei die aktuelle Pinselposition als Schnittpunkt bestimmt wird
K	Bildschirm löschen
m	Vergrößerung an/aus
p	Farbpalette
q	Kurvenlinien
r	Ungefülltes Vieleck
R	Gefülltes Vieleck
s	Punktiertes Freihandzeichnen
t	Texteingabe
u	Letzte Änderung zurücknehmen
v	Gerade Linie
>	Vergrößerung – Maßstab größer
<	Vergrößerung – Maßstab kleiner
,	Farbwahl (Pick)

Die Zeichenmodi

F1	Muster
F2	Farbe
F3	Stempel
F4	Schmieren
F5	Schatten
F6	Mischen
F7	Rollen
F8	Weich

Animation und Tastatur

1	Ein Frame zurück
2	Ein Frame vorwärts
3	Gehe zu Frame Nr.
4	Kontinuierliches Abspielen
5	Einmaliges Abspielen
6	Animation vorwärts und rückwärts (Ping Pong) abspielen

Brushbefehle auf der Tastatur

Z	Dehnen
h	Größe halbieren
H	Größe verdoppeln
x	Horizontal klappen
y	Vertikal klappen
z	90 Grad drehen
X	In horizontaler Richtung verdoppeln
Y	In vertikaler Richtung verdoppeln



Sebastian Ritter

Quadrant 500

Alles auf einen Blick – Sieh zu, aber mach hin!

Bei unserem AmigaBASIC-Listing Quadrant 500 handelt es sich um ein Spiel, das eine schnelle Auffassungsgabe verlangt. Es gilt, innerhalb einer sich von Level zu Level dezimierenden Zeitspanne, eine bestimmte Farbkombination zu finden.

Nach dem Starten des Programms erscheint das Titelbild. Durch Betätigen der Feuertaste am Joystick gelangt man in die Highscore-Liste, die als separate Datei auf Diskette abgespeichert wird. Das Abspeichern dieser Liste erfolgt automatisch durch das Programm. Wundern Sie sich also nicht, wenn irgendwann einmal eine Datei namens Hiscore auf Ihrer Diskette abgelegt wird. – Das ist alles ganz normal.

Aufmerksamkeit ist alles

Mit der Tastenfunktion <Alt S> läßt sich diese Highscore-Liste übrigens abspeichern, während durch gleichzeitiges Drücken der Tasten <Alt L> eine bereits abgespeicherte Liste eingeladen wird. Nun muß die Anzahl der Farben im Spiel eingegeben werden (Anm. der Red.: je mehr, desto leichter).

Zu Beginn des Spieles sieht man am rechten Bildschirmrand ein in zwölf mal zwölf Einheiten unterteiltes Quadrat. Jedes Kästchen dieser Fläche besitzt eine bestimmte Farbe. Am linken Bildrand ist ein drei mal drei Einheiten großes Quadrat, das einen Ausschnitt aus dem großen Quadrat anzeigt. Aufgabe des Spielers ist es nun, diese Farbsequenz des Ausschnitts mittels Cursor, der mit dem Joystick gesteuert wird, herauszusuchen und durch Drücken der Feuertaste zu fixieren. Das wäre ja recht einfach, wenn nicht...

Wer sucht, der findet

...ja, wenn der Spieler unbegrenzte Zeit für diese Aktion hätte. Dem ist jedoch nicht so, denn nach 26 Sekunden (im ersten Level) ist "Ende im Gelände".
Wurde ein falscher Ausschnitt gewählt oder das Zeitlimit überschritten, so ist das Spiel beendet. Hat der Spieler währenddessen den korrekten Ausschnitt gefunden, wird er mit einem Bonus in Höhe der verbliebenen Zeit, multipliziert mit dem Faktor zehn, belohnt. Bei entsprechend hoher Punktzahl ist ein Highscore-Eintrag (maximal 16 Zeichen) möglich.

(br)

Wichtig!!!

Die Zeilennummern am Anfang der Programmzeilen dienen nur der besseren Übersicht und sollten im BASIC-Editor nicht mit eingegeben werden. Sie können zur Eingabe auch den Checksummer CheckBAS aus Ausgabe 1/90 benutzen.

Listings

```
000:998 REM ************************************************
001:154 REM *                 QUADRANT 500                 *
002:182 REM *                    Spiel                     *
003:916 REM *     (c) 1990 by S.Ritter & AMIGA DOS         *
004:612 REM *              Sprache: AmigaBasic             *
005:998 REM ************************************************
006:000
007:577  DIM ta(15),feld(12,12),kfel(3,3),pun%(14),nam$(14),
         hi$(16)
008:706  FOR i%=1 TO 14
009:120    pun%(i%)=16000-i%*1000
010:018    nam$(i%)="Sebastian Ritter"
011:823  NEXT i%
012:288  RANDOMIZE TIMER
013:053  SCREEN 2,320,200,5,1
014:506  WINDOW 2,"Quadrant 500 (c) S. Ritter & AMIGA DOS",(
         0,0)-(311,178),0,2
015:800 intro:
016:691  FOR i%=0 TO 31
017:064    PALETTE i%,0,0,0
018:823  NEXT i%
019:765  CALL titel (ta(),a$)
020:182  sc%=0
```

Listing: Quadrant 500

```
021:543  lev%=1
022:627  CALL hiscoretabelle (pun%(),nam$(),sc%,hi$())
023:304  CALL spfarben
024:399  CALL farbenzahl (fa%)
025:936  spielfeldaufbau:
026:928  CALL spielfeld
027:704  FOR i%=1 TO 12
028:440    FOR i1%=1 TO 12
029:715      zf%=RND*fa%+1
030:601      feld(i%,i1%)=zf%
031:506      LINE((i%-1)*14+10,(i1%-1)*14+10)-((i%-1)*14+16,
        (i1%-1)*14+16),zf%+14,BF
032:707    NEXT i1%
033:823  NEXT i%
034:179  z1%=INT(RND*10)+1
035:947  z2%=INT(RND*10)+1
036:329  FOR i%=1 TO 3
037:697    FOR i1%=1 TO 3
038:247      zf%=feld(z1%+i%-1,z2%+i1%-1)
039:249      kfel(i%,i1%)=zf%
040:034      LINE((i%-1)*14+268,(i1%-1)*14+10)-((i%-1)*14+27
        4,(i1%-1)*14+16),zf%+14,BF
041:707    NEXT i1%
042:823  NEXT i%
043:732  COLOR 28,4
044:855  LOCATE 14,23
```

Listing: Quadrant 500

```
PRINT USING"  Punkte:\       \";STR$(sc%)
LOCATE 21,23
PRINT USING"  Level :\       \";STR$(lev%)
LOCATE 7,23
PRINT "  Zeit:";SPACE$(4)
x%=6
y%=6
zei%=27-lev%
IF zei%<8 THEN zei%=8
ON TIMER(1)GOSUB zeitkontrolle
LINE (x%,y%)-(x%+42,y%+42),28,b
WHILE INKEY$<>""
WEND
hauptschleife:
FOR i=1 TO 50:NEXT
TIMER ON
IF zei%<=0 GOTO levelende
fsv=STICK(2):mz=STICK(3):os=STRIG(3)
IF fsv=0 AND mz=0 AND os=0 GOTO hauptschleife
IF os=-1 GOTO levelende
h%=0
v%=0
IF mz=-1 AND y%>6 THEN v%=-14
IF mz=1 AND y%<132 THEN v%=14
IF fsv=1 AND x%<132 THEN h%=14
IF fsv=-1 AND x%>6 THEN h%=-14
IF h%=0 AND v%=0 GOTO hauptschleife
LINE (x%,y%)-(x%+42,y%+42),29,b
y%=y%+v%
x%=x%+h%
LINE (x%,y%)-(x%+42,y%+42),28,b
GOTO hauptschleife

SUB farben (ta()) STATIC
 i%=1
 FOR t=.9 TO .2 STEP-.1
   ta(i%)=t
   PALETTE i%,t,t,t
   i%=i%+1
 NEXT t
 FOR t=.1 TO .9 STEP .1
   ta(i%)=t
   PALETTE i%,t,t,t
   i%=i%+1
 NEXT t
 PALETTE 30,1,0,1
 PALETTE 31,.33,.87,1
END SUB

SUB spfarben STATIC
 FOR i%=1 TO 14
   PALETTE i%,.25+(16-i%)*.05,0,0
 NEXT i%
 RESTORE farbdatas1
 FOR i%=15 TO 30
   READ a,b,c
   PALETTE i%,a,b,c
 NEXT
 PALETTE 31,0,.93,.87
END SUB

SUB titel (ta(),a$) STATIC
 CLS
 f%=0
 WHILE f%<178
   FOR i%=1 TO 13
     LINE (0,f%)-(312,f%),i%
     LINE (0,f%+1)-(312,f%+1),i%
     f%=f%+2
   NEXT i%
 WEND
 LOCATE 7,14
 COLOR 31,30
 PRINT "Quadrant 500"
 LOCATE 17,5
 PRINT "1990 by S. Ritter & AMIGA DOS"
 CALL farben (ta())
   WHILE sgr=0
   FOR i%=1 TO 13
     FOR i1%=1 TO 13
       IF STRIG(3)=-1 THEN EXIT SUB
       t=ta((i%+i1%)MOD 13 +2)
       PALETTE i1%,t,t,t
     NEXT i1%
   NEXT i%
   WEND
END SUB

SUB spielfeld STATIC
 f%=0
 WHILE f%<178
     FOR i%=1 TO 14
       LINE (0,f%)-(312,f%),i%
       f%=f%+1
     NEXT i%
   FOR i%=13 TO 1 STEP-1
     LINE (0,f%)-(312,f%),i%
     f%=f%+1
   NEXT i%
 WEND
 FOR i%=6 TO 174 STEP 14
   LINE (6,i%)-(173,i%),29
   LINE (i%,6)-(i%,174),29
 NEXT i%
 FOR i%=6 TO 48 STEP 14
   LINE(264,i%)-(306,i%),29
   LINE(i%+258,6)-(i%+258,48),29
```

Listing: Quadrant 500

```
 NEXT i%
END SUB

SUB hiscoretabelle (pun%(),nam$(),sc%,hi$()) STATIC
 FOR i%=1 TO 15
   PALETTE i%+14,.7,.09+i%/100*6,0
 NEXT i%
 a%=0
 WHILE a%<178
   FOR i%=29 TO 15 STEP-1
     LINE (0,a%)-(312,a%),i%
     LINE (0,a%+1)-(312,a%+1),i%
     a%=a%+2
   NEXT i%
   FOR i%=16 TO 28
     LINE (0,a%)-(312,a%),i%
     LINE (0,a%+1)-(312,a%+1),i%
     a%=a%+2
   NEXT i%
 WEND

namensausgabe:
 PALETTE 31,.5,0,0
 COLOR 15,22
 LOCATE 2,15
 PRINT "Highscores"
 FOR i%=15 TO 28
   COLOR 31,i%
   LOCATE i%-10,7
   PRINT SPACE$(27)
   LOCATE i%-10,8
   PRINT nam$(i%-14)
   LOCATE i%-10,25
   PRINT USING"########";pun%(i%-14)
 NEXT i%
 IF sc%<pun%(14)GOTO tastendruck
 COLOR 31,29
 LOCATE 22,1
 PRINT SPACE$(39);
 LOCATE 22,1
 PRINT "Bitte Namen eingeben:";
 FOR i%=1 TO 16
   hi$(i%)=INPUT$(1)
   IF hi$(i%)=CHR$(13)GOTO einordnen
   IF hi$(i%)=CHR$(31)AND i%>0 THEN i%=i%-2
   IF i%<0 THEN i%=0
   IF hi$(i%)<CHR$(32) OR hi$(i%)>CHR$(122) THEN i%=i%-1
   IF i%<0 THEN i%=0
   LOCATE 22,22+i%
   PRINT hi$(i%);
 NEXT i%

einordnen:
 nam$(14)=""
 FOR i%=1 TO 16
   nam$(14)=nam$(14)+hi$(i%)
 NEXT i%
 pun%(14)=sc%
 FOR i%=1 TO 13
   p%=i%
   FOR i1%=i%+1 TO 14
     IF pun%(p%)<pun%(i1%)THEN p%=i1%
   NEXT i1%
   pun%(0)=pun%(i%)
   pun%(i%)=pun%(p%)
   pun%(p%)=pun%(0)
   nam$(0)=nam$(i%)
   nam$(i%)=nam$(p%)
   nam$(p%)=nam$(0)
 NEXT i%
 sc%=0
 LOCATE 22,1
 PRINT SPACE$(39);
GOTO namensausgabe

tastendruck:
 WHILE INKEY$<>"":WEND
 a$=""
 WHILE a$="" AND STRIG(3)=0
 a$=INKEY$
 WEND
 l%=0
 IF a$=CHR$(223)THEN GOSUB hiscoresave
 IF a$=CHR$(163)THEN GOSUB hiscoreload
 IF l%=1 THEN GOTO namensausgabe
 CLS
 EXIT SUB
 hiscoresave:
  OPEN"hiscore"FOR OUTPUT AS #1
  FOR i%=1 TO 14
    PRINT #1,pun%(i%)
    PRINT #1,nam$(i%)
  NEXT i%
  CLOSE#1
 RETURN

 hiscoreload:
  l%=1
  OPEN"hiscore"FOR INPUT AS #1
  FOR i%=1 TO 14
    INPUT#1,pun%(i%)
    INPUT#1,nam$(i%)
  NEXT i%
  CLOSE#1
 RETURN
```

Listing: Quadrant 500

```
257:000
258:568 END SUB
259:000
260:293 SUB farbenzahl (fa%) STATIC
261:282  lin=0
262:668  FOR i%=0 TO 312 STEP 20
263:922    LINE (i%,0)-(i%+19,180),i%/20+14,BF
264:823  NEXT i%
265:000
266:030 fragebox:
267:930  LINE (0,80)-(312,100),1,BF
268:349  COLOR 9,1
269:396  LOCATE 11,8
270:316  PRINT "Wieviele Farben (2-16)?"
271:843  LOCATE 12,19
272:855  INPUT "",fa%
273:398  IF fa%<2 OR fa%>16 GOTO fragebox
274:487  fa%=fa%-1
275:568 END SUB
276:000
277:628 levelende:
278:174  TIMER OFF
279:530  k%=(x%+8)/14
280:706  l%=(y%+8)/14
281:076  FOR i%=k% TO k%+2
282:100    FOR il%=l% TO l%+2
283:552      IF feld(i%,il%)<>kfel(i%-k%+1,il%-l%+1)GOTO ver
        loren
284:707    NEXT il%
285:823  NEXT i%
286:253  sc%=sc%+zei%*10*lev%
287:357  GOSUB punkteausdruckroutine
288:315  lev%=lev%+1
289:836  WHILE STRIG(3)=0:WEND
290:047 GOTO spielfeldaufbau
291:000
292:938 verloren:
293:902  LINE (x%,y%)-(x%+42,y%+42),29,b
294:058  LINE (z1%*14-8,z2%*14-8)-(z1%*14+34,z2%*14+34),28,b
295:516  COLOR 29,20
296:826  LOCATE 10,26
297:253  PRINT SPACE$(10)
298:523  COLOR 29,19
299:834  LOCATE 11,26
300:442  PRINT " Verloren "
301:518  COLOR 29,22
```

Listing: Quadrant 500

```
302:842  LOCATE 12,26
303:253  PRINT SPACE$(10)
304:408  FOR i%=600 TO 200 STEP-20
305:689    FOR il%=0 TO 3
306:867      SOUND i%+il%*15,2,255,0+il%
307:707    NEXT il%
308:823  NEXT i%
309:836  WHILE STRIG(3)=0:WEND
310:503  CLS
311:627  CALL hiscoretabelle (pun%(),nam$(),sc%,hi$())
312:563 GOTO intro
313:000
314:628 punkteausdruckroutine:
315:856  LOCATE 14,32
316:047  PRINT USING"\      \";STR$(sc%)
317:848  LOCATE 21,32
318:227  PRINT USING"\      \";STR$(lev%)
319:702  COLOR 13,2
320:193  LOCATE 7,5
321:115  PRINT SPACE$(9)
322:706  COLOR 13,6
323:197  LOCATE 8,5
324:966  PRINT " Richtig "
325:690  COLOR 2,10
326:201  LOCATE 9,5
327:115  PRINT SPACE$(9)
328:771  FOR i%=255 TO 0 STEP-15
329:388    SOUND 262,2,i%,0
330:877    SOUND 330,2,i%,1
331:670    SOUND 392,2,i%,2
332:031    SOUND 523,2,i%,3
333:823  NEXT i%
334:982 RETURN
335:000
336:404 zeitkontrolle:
337:671  zei%=zei%-1
338:431  LOCATE 7,31
339:363  PRINT USING"##";zei%
340:982 RETURN
341:000
342:184 farbdatasl:
343:943  DATA .4,.6,1,0,0,1,1,.6,.67,.3,.3,.3
344:551  DATA 0,1,0,.4,.7,.5,1,.82,0,.1,.5,.15
345:173  DATA 1,0,1,1,.6,1,.3,.4,.5,.8,.6,.53
346:545  DATA .5,.3,.7,1,1,1,0,0,0,.5,.5,.5
```

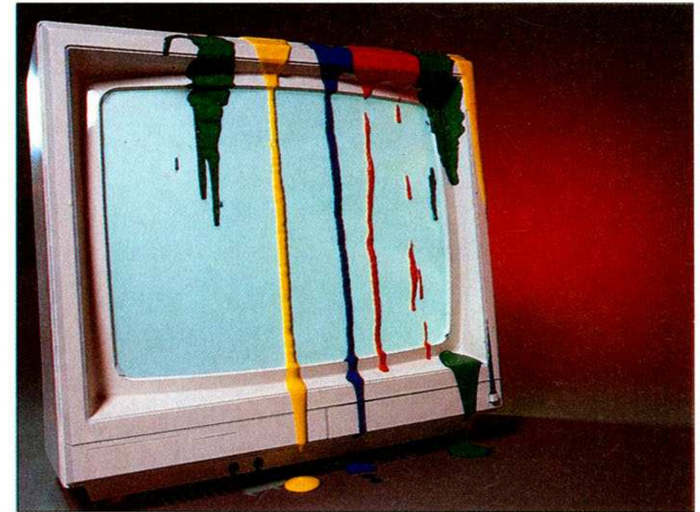
Listing: Quadrant 500

Monika Schmid

Meister Klecksel unterwegs

Farben ins CLI gebracht

Vorbei sind die Zeiten, wo der Kommandozeilen-Interpreter als kümmerliches graues Mäuschen sein Dasein ohne jegliche Farbtupfer fristete. Ein gutes AmigaDOS-Handbuch, ein wenig Überlegung, und schon kann's losgehen. Tricksen wir ein wenig mit dem CLI herum.



In Kürze noch einiges zum CLI: Möchte man sich nicht auf die Ebene der Workbench begeben, sondern einen noch etwas innigeren Kontakt zum Rechner herstellen, begibt man sich auf die Ebene des CLI. Direkt nach dem Start des Amiga kommt man zum ersten Mal mit dem CLI, CommandLineInterpreter, in Berührung. Das erste CLI-Fenster gibt die Meldungen des DOS aus. CLI stellt also die direkte Beziehung zwischen dem Anwender, der seine Befehle eingibt, dem DOS, das diese Befehle maschinengerecht aufbereitet und dem Rechner selbst, der diese Befehle verarbeitet, her.

Das CLI selbst ist an und für sich recht farblos – im wahrsten Sinne des Wortes. Eigentlich ist es die einfachste Sache der Welt, Farben in den CLI zu bringen. Ein kleiner Trick hilft da weiter. Man muß nur die Steuersequenzen aus dem AmigaDOS-Handbuch eintippen, RETURN drücken und das "unknown command" über sich ergehen lassen. Schon kann man im CLI mit der gewünschten Vordergrund- oder Hintergrundfarbe weiterschreiben. Ebenso lassen sich die Styles normal, fett, unterstrichen und kursiv darstellen, wobei kursiv erst dann richtig zur Geltung kommt, wenn der Transparent-Modus eingestellt ist.

Einige Sequenzen, wie man beispielweise Styles verändert: Setzt man

- ESC[0m RETURN

ein, kann man alle eingestellten Werte wieder zurücksetzen, oder mit

- ESC[3m RETURN

eine kursive Ausgabe erreichen oder

- ESC[4m RETURN

den Ausgabetext unterstrichen darstellen oder

- ESC[1m RETURN

den Ausgabetext in fetter Schrift darstellen.

Mit ESC ist dabei nicht der Text ESC gemeint, sondern die Escape-Taste. Ebenso ist mit RETURN nicht der Text "RETURN" gemeint, sondern die RETURN-Taste. Diese Informationen können Sie dem AmigaDOS-Handbuch (ganz hinten bei den Drucker-Steuersequenzen) entnehmen. Für Amiga-Einsteiger ist es sicherlich nicht bekannt, daß man diese Steuersequenzen direkt ins CLI-Fenster eintippen kann, um den Farb- oder Stilwechsel herbeizuführen. Die anderen Steuersequenzen, die weder Farbe noch Stil betreffen, funktionieren im CLI allerdings nicht, soweit ich das ausprobiert habe.

Aus einigen "Try and Error"-Erfahrungen entstand nun ein ein kleines Assembler-Programm, das nach der gewünschten Farbe fragt und diese dann einstellt.

Das Ganze wurde mit dem DevPac-Assembler erstellt.

(vb)

Literatur

Commodore Amiga Inc., AmigaDOS 1.3, ISBN 3-89090-802-0, Markt&Technik Verlag, 1989, München

Listings

```
; ****************************************************
; *                   bunt.asm                       *
; *     Dieses Programm bringt Farben in den CLI     *
; *        (c) 1990 M. Schmid & AMIGA DOS            *
; *     Sprache: Devpac Assembler                    *
; ****************************************************

include "sys:exec/exec_lib.i"          ; ggfs. anpassen
include "sys:libraries7dos_lib.i"      ; ggfs. anpassen

         movem.l   d0-d7/a0-a6,-(sp) ; Register retten
         lea       dosname,a1        ; Lib.Name nach A1
         move.l    #0,d0             ; Version nach D0
         callexec  OpenLibrary
         tst.l     d0
         beq       Fehler
         move.l    d0,_Dosbase      ;Speichern Dos-Zeiger

         calldos   Output           ;suche Filehandler
des aktuellen Fensters
         tst.l     d0
         beq       aus
         move.l    d0,Aktuellesfenster

Fragen1:
         move.l    #0,Antwort1
         move.l    Aktuellesfenster,d1
         move.l    #Bitte1,d2       ;Vordergrundfarbe ?
         move.l    #25,d3
         calldos   write

Eingabe1:
         move.l    Aktuellesfenster,d1
         move.l    #1,d2            ; auf Eingabe warten
         calldos   WaitForChar
         tst.l     d0
         beq       Eingabe1

Eingabe2:
         move.l    Aktuellesfenster,d1
         clr.l     d0
```

Listing: bunt.asm

```
         move.l    #Antwort1,d2
;dahinein sollen die Zeichen geschrieben werden
         move.l    #70,d3
         calldos   read
; 70 Zeichen aus Tastatur in den Datenpuffer schreiben
         cmpi.l    #-1,d0           ; Ende bei Fehler
         beq       aus
                                    ; Anzahl Zeichen bis
                                    ; zum RETURN gelesen
         cmpi.l    #2,d0            ; = 2 Zeichen
         beq       recht1
         bsr       zulang           ; oder zu kurz
         bra       Fragen1
recht1:
         move.l    #Antwort1,a0
         move.b    (a0),d7          ;Ziffer holen
         cmpi.b    #"0",d7
         beq       recht2
         cmpi.b    #"1",d7
         beq       recht2
         cmpi.b    #"2",d7
         beq       recht2
         cmpi.b    #"3",d7
         beq       recht2
         bsr       wiebitte


         bra       Fragen1
recht2:

Fragen2:
         move.l    #0,Antwort2
         move.l    Aktuellesfenster,d1
         move.l    #Bitte2,d2
         move.l    #25,d3
         calldos   write

Eingabe3:
         move.l    Aktuellesfenster,d1
         move.l    #1,d2            ; auf Eingabe warten
         calldos   WaitForChar
         tst.l     d0
         beq       Eingabe3

Eingabe4:
```

Listing: bunt.asm

```
        move.l     Aktuellesfenster,d1
        clr.l      d0
        move.l     #Antwort2,d2      ; Zeichen werden da
; hinein geschrieben
        move.l     #70,d3
        calldos    read
; 70 Zeichen von Tastatur in  Datenpuffer schreiben
        cmpi.l     #-1,d0            ; Ende bei Fehler
        beq        aus
; so viele Zeichen konnten bis zum
; RETURN gelesen werden

        cmpi.l     #2,d0             ; = 2 Zeichen
        beq        recht3
        bsr        zulang            ; oder zu kurz
        bra        Fragen2
recht3:
        move.l     #Antwort2,a0
        move.b     (a0),d7           ; Ziffer holen
        cmpi.b     #"0",d7
        beq        recht4
        cmpi.b     #"1",d7
        beq        recht4
        cmpi.b     #"2",d7
        beq        recht4
        cmpi.b     #"3",d7
        beq        recht4
        bsr        wiebitte
        bra        Fragen2
recht4:
vergleichen:
        move.l     #Antwort1,a1
        move.b     (a1),d1
        move.l     #Antwort2,a2
        move.b     (a2),d2
        cmp.b      d1,d2
        beq        blind

Setzen0:
        move.l     #Antwort1,a2
        move.b     (a2),d2
        cmpi.b     #"0",d2
        bne        setzen1
        move.l     Aktuellesfenster,d1
        move.l     #SetAPen0,d2
        move.l     #6,d3
        calldos    write
        bra        Hinter0

Setzen1:
        move.l     #Antwort1,a2
        move.b     (a2),d2
        cmpi.b     #"1",d2
        bne        Setzen2
        move.l     Aktuellesfenster,d1
        move.l     #SetAPen1,d2
        move.l     #6,d3
        calldos    write
        bra        Hinter0
Setzen2:
        move.l     #Antwort1,a2
        move.b     (a2),d2
        cmpi.b     #"2",d2
        bne        setzen3
        move.l     Aktuellesfenster,d1
        move.l     #SetApen2,d2
        move.l     #6,d3
        calldos    write
        bra        Hinter0
Setzen3:
        move.l     #Antwort1,a2
        move.b     (a2),d2
        cmpi.b     #"3",d2
        bne        Hinter0     ; eigentlich unmöglich
        Move.l     Aktuellesfenster,d1
        move.l     #SetAPen3,d2
        move.l     #6,d3
        calldos    write

Hinter0:
        move.l     #Antwort2,a2
        move.b     (a2),d2
        cmpi.b     #"0",d2
        bne        Hinter1
        move.l     Aktuellesfenster,d1
        move.l     #SetBPen0,d2
        move.l     #8,d3
        calldos    write
        bra        Ende
Hinter1:
        move.l     #Antwort2,a2
        move.b     (a2),d2
        cmpi.b     #"1",d2
        bne        Hinter2
        move.l     Aktuellesfenster,d1
        move.l     #SetBPen1,d2
        move.l     #8,d3
        calldos    write
        bra        Ende
Hinter2:
        move.l     #Antwort2,a2
        move.b     (a2),d2
        cmpi.b     #"2",d2
        bne        Hinter3
        move.l     Aktuellesfenster,d1
        move.l     #SetBPen2,d2
        move.l     #8,d3
        calldos    write
        bra        Ende
Hinter3:
        move.l     #Antwort2,a2
        move.b     (a2),d2
        cmpi.b     #"3",d2
        bne        Ende   ; darf eigentlich nicht sein
        move.l     Aktuellesfenster,d1

        move.l     #SetBPen3,d2
        move.l     #8,d3
        calldos    write
Ende:
; ** hier evtl. Probetext zu Testzwecken ausgeben ****
```

Listing: bunt.asm

```
aus:
        move.l     _Dosbase,a1
        callexec   CloseLibrary

Fehler:
        movem.l    (sp)+,d0-d7/a0-a6 ; Register zurück
        rts                          ; Rück-> Programmende


; ************* Hauptprogrammfragmente ***************

blind:
        move.l     #0,Antwort3
Fragen3:
        move.l     Aktuellesfenster,d1
        move.l     #Unsichtbar,d2
        move.l     #46,d3
        calldos    write

Eingabe5:
        move.l     Aktuellesfenster,d1
        move.l     #1,d2         ; auf Eingabe warten
        calldos    WaitForChar
        tst.l      d0
        beq        Eingabe5

; **************** Dateinamen eingegeben wurde *******

Eingabe6:
        move.l     Aktuellesfenster,d1
        clr.l      d0
        move.l     #Antwort3,d2
;dahinein sollen die Zeichen geschrieben werden
        move.l     #70,d3
        calldos    read
; 70 Zeichen von der Tastatur in Datenpuffer schreiben
        cmpi.l     #-1,d0        ; Ende bei Fehler
        beq        aus
; so viele Ziffern konnten bis
; zum RETURN gelesen werden
        cmpi.l     #2,d0         ; = 2 Zeichen
        beq        recht5
        bsr        zulang        ;oder zu kurz
        bra        Fragen3
recht5:
        move.l     #Antwort3,a0
        move.b     (a0),d7       ;Ziffer holen
        cmpi.b     #"j",d7
        beq        recht6
        cmpi.b     #"n",d7
        beq        recht6
        bsr        wiebitte
        bra        Fragen3
recht6:
        cmpi.b     #"n",d7
        beq        Fragen1
        bra        setzen0


;  ***********  Unterprogramme  *****************


zulang:
        move.l     Aktuellesfenster,d1   ; Return
        move.l     #Lang,d2
        move.l     #1,d3
        calldos    write

wiebitte:
        move.l     Aktuellesfenster,d1
        move.l     #Bahnhof,d2
        move.l     #25,d3
        calldos    write
        rts

; ************** Definitionen **********************

dosname:
        dc.b       "dos.library",0
        even
_Dosbase:
        dc.l 0
Antwort1:
        ds.b 80                  ; max. 80 Bytes
Antwort2:
        ds.b 80                  ; max. 80 Bytes
Antwort3:
        ds.b 80                  ; max. 80 Bytes
Bitte1:
        dc.b       "Vordergrundfarbe ? (0-3) ",0
        even
Bitte2:
        dc.b       "Hintergrundfarbe ? (0-3) ",0
        even
Aktuellesfenster:
        dc.l       0
        even
Lang:
        dc.b       $0A,0
        even
Bahnhof:
        dc.b       "Das verstehe ich nicht !",10,0
        even
Unsichtbar:
        dc.b       "Wollen Sie wirklich unsichtbar we
rden (j/n) ? ",0
        even
SetAPen0:
        dc.b       27,"[30m",0  ; Penfarbe auf 0
SetAPen2:
        dc.b       27,"[32m",0  ; Penfarbe 2
SetAPen3:
        dc.b       27,"[33m",0  ; Penfarbe 1
SetBPen1:
        dc.b       27,"[41m",10,0,0 ; Paper 1
SetBPen2:
        dc.b       27,"[42m",10,0,0 ; Paper 2
SetBPen3:
        dc.b       27,"[43m",10,0,0 ; Paper 3
SetAPen1:
        dc.b       27,"[31m",0  ; Penfarbe 1
SetBPen0:
        dc.b       27,"[40m",10,0,0 ; Paper 0

; ********* Ende von bunt.asm ***********************
```

Listing: bunt.asm

AMIGA DOS

Endlich keine Listings mehr abtippen!

Nicht bei allen Programmen ist es mit drei Zeilen getan – gute Routinen und praktische Funktionen brauchen ihren Platz. Und bisweilen lassen sich auch lange Datenblöcke nicht vermeiden, ganz zu schweigen von Hexdumps und Assemblerlistings.
Schonen Sie Ihre Augen und schlagen Sie sich nicht die Nacht mit Abtippen um die Ohren. – Auf der Databox zum Amiga DOS-Heft finden Sie alle Listings als ASCII-File: passend für jeden Texteditor, den Amiga-BASIC-Interpreter, Makro-Assembler oder einen Compiler für C und Modula-2.

Alle Programme sofort nutzen

Da ist er nun endlich – der Trick oder das Programm, auf das Sie schon so lange gewartet haben! Zu allem Unglück ist das Listing aber in Modula-2 oder C, jedenfalls in einer Sprache, zu der Sie keinen Compiler haben, um ein lauffähiges Programm herzustellen.

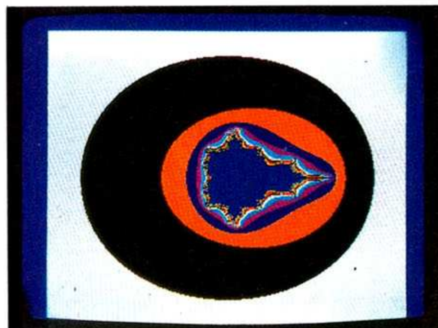
Auch in diesem Fall hilft Ihnen die Databox von Amiga DOS aus der Patsche: Neben den Quelltexten im ASCII-Format finden Sie jeweils auch das fertige, lauffähige Programm. Sie brauchen es nur von der Databox-Diskette aus zu starten.

DATA

Keinen Ärger mehr mit Tippfehlern

Wer kennt das nicht, wenn das Programm nach dem Eintippen nicht läuft oder der Rechner gar abstürzt. Besonders gemein sind auch Fehler, die erst nach Wochen bei einer bisher nicht gebrauchten Funktion zu Tage treten, oder wenn der Druckfehlerteufel am Werke war.

Zermartern Sie sich nicht den Kopf, bis Sie die falsche Zahl im Datafeld gefunden haben. – Alle Dateien auf der Databox zur Amiga DOS sind vom Autor und der Redaktion auf Fehlerfreiheit geprüft und im dazugehörigen System "probegelaufen".



Zaubern Sie sich mit unserem Fraktal-Generator in eine faszinierende Grafikwelt



Um sich bei dem AmigaBASIC-Listing Quadrant 500 in die HighScore-Liste eintragen zu können, bedarf es fast Adleraugen



Der Faszination fraktaler Grafiken kann sich kaum jemand entziehen. Wir präsentieren Ihnen den schnellsten Fraktalgenerator für den Amiga, der je abgedruckt wurde.

Peter Woop

Faszination Fraktal

Wer Mandelbrotfiguren berechnen will, ohne darauf mehrere Stunden warten zu wollen, muß sich auch auf einem nicht gerade langsamen Rechner wie dem Amiga etwas einfallen lassen. Unser Listing zeigt, wie's schnell und benutzerfreundlich geht.

Als Programmiersprache wurde einmal mehr Assembler gewählt, da nur so einige der elementaren Ideen dieses Programms zu verwirklichen waren. Allerdings sieht man auch an der Länge des Listings, daß Assemblerprogramme (zumindest was den Quellcode betrifft) meist um einiges länger sind als in Hochsprachen entwickelte. Aber was tut man nicht alles, um etwas weniger lange warten zu müssen.

Die Funktionen

Doch nun zur Funktionsweise: Das Programm berechnet die zum Setzen der Punkte benötigte Adresse nicht jedesmal neu, sondern füllt den Bildschirm von oben nach unten. Dabei wird einfach das entsprechende Adreßregister, das auf den Speicher zeigt, fortlaufend erhöht. Auch muß kein unnötiger Aufwand damit betrieben werden, ein Wort aus dem Speicher zu holen und mit den zu setzenden Bits zu verknüpfen, um es dann wieder zu schreiben. Statt dessen lassen sich die 16 Bits eines Wortes per Oder-Befehl direkt mit dem Speicher verknüpfen, da jeder Punkt nur einmal gesetzt wird und somit vor dem Setzen gleich null ist.

Jetzt kommt Farbe ins Spiel

Auch bei der Farbwahl wurde optimiert: Hier wird für jede der 16 möglichen Farben in eine eigene Routine verzweigt, die nur die Bitplanes beschreibt, bei denen dies wirklich notwendig ist. Da bei diesem Verfahren die Bitplanes unbedingt hintereinander liegen müssen, wird nach dem Programmstart ein zusammenhängender Speicherblock reserviert, der den eigentlichen, möglicherweise auseinandergerissenen, Speicher ersetzt. Eine weitere Geschwindigkeitssteigerung läßt sich durch eine bestimmte Eigenschaft der Mandelbrotmenge erreichen: Liegen nämlich zwei Punkte nur nahe genug beieinander, so liefern sie auch dasselbe Ergebnis. Das Programm testet zunächst, ob in einer Entfernung von acht Bildschirmpunkten immer noch derselbe Funktionsswert geliefert wird. Ist dies der Fall, werden acht Punkte der entsprechenden Farbe gesetzt, ansonsten reduziert sich die Schrittweite auf die Hälfte. Im schlechtesten Fall werden alle Punkte wieder einzeln berechnet. Allerdings hat auch dieses Verfahren einen Haken, denn sollte sich der Funktionswert zwischen zwei Punkten mit demselben Wert kurz ändern, so bleibt dies unberücksichtigt. Obwohl dieses Phänomen im Verhältnis zum gesamten Bildschirm nur recht selten auftritt, trägt es nicht gerade zur Ansehnlichkeit der Bilder bei. Aus diesem Grund wurde eine weitere Funktion implementiert, mit der der Benutzer bestimmte Teile eines Bildes ohne diese Sprungtechnik, also auch ohne mögliche Fehler, nachträglich neu berechnen kann.

Präzision mit 32 Bit

Durch die Verwendung einer weiteren Iterationsroutine mit geringerer Rechengenauigkeit als der im Programm verwendeten (32 Bit) ließe sich die Geschwindigkeit noch weiter steigern. Weil aber mit einer derartigen Routine nicht viel mehr als die Berechnung der Urfigur möglich ist, wurde diese Idee schnell wieder verworfen. Auch wenn sich die Programmierung kompliziert anhören mag, die Bedienung des Programms ist es keineswegs: Nach dem Start erscheint ein Fenster auf dem CLI-Bildschirm, das alle erforderlichen Bedienelemente enthält. Dort findet man unter anderem einige String-Gadgets, in die man nach der Aktivierung per Mausklick die zur Berechnung des Mandelbrotbildes nötigen Werte wie Koordinaten und Rechentiefe eingeben kann. Dabei ist das Intervall des Realteils (x-Richtung) und des Imaginärteils (y-Richtung) in derselben Form anzugeben, in der man die Anfangswerte (jeweils von -3,5 bis + 3,5) nach dem Start vorfindet. Dies ist erforderlich, da eine komplexe Überprüfung auf Eingabefehler das Programm noch weiter in die Länge gezogen hätte. Allerdings braucht kein führendes Pluszeichen angegeben zu werden, ebenso wie Nullen am Ende der Zahl wegfallen können. Die Rechentiefe gibt die Anzahl der Iterationen an, nach denen spätestens abgebrochen werden soll. Mit dem Anklicken von "Bild berechnen" läßt sich schließlich das Fraktal errechnen, das übrigens auf einem zweiten Bildschirm angezeigt wird. Obwohl das Programm nach jeder Berechnung wieder zum CLI-Bildschirm umschaltet, läßt sich natürlich jederzeit mit der linken Amiga-Taste und M oder N zwischen den Screens hin- und herspringen.

Bei "Ausschnitt" erscheint ein Fadenkreuz, mit dem per Maus der Bereich des Bildes markiert werden kann, der später vergrößert berechnet werden soll.

Im Fadenkreuz: fraktale Grafiken

Dieser Vorgang läßt sich übrigens beliebig oft wiederholen, da die Koordinaten erst dann endgültig übernommen werden, wenn ein Bild berechnet wird. "Polieren" erlaubt in ähnlicher Weise die Auswahl eines Bildbereiches. Allerdings bleibt hier das berechnete Bild bestehen, und nur dieser Bereich wird, wie weiter oben schon erwähnt, nochmals Punkt für Punkt errechnet. Schließlich gestattet die Funktion "Bild speichern" das Abspeichern eines errechneten Fraktals auf Diskette, wobei das IFF-Format Verwendung findet. Allerdings wurde (auch hier aus Platzgründen) auf den Pack-Algorithmus verzichtet, so daß das Bild unkomprimiert gespeichert wird. Hierbei ist zu berücksichtigen, daß jedes Bild etwa 41 kByte in Anspruch nimmt und natürlich entsprechend viel Platz auf der Diskette vorhanden sein muß. Ansonsten bleibt uns nur noch, Ihnen viel Spaß beim Errechnen der Bilder zu wünschen. Im übrigen sollte es kein allzu großes Problem darstellen, nicht implementierte Funktionen wie Packer, Farbeinsteller oder Überprüfung von Eingabefehlern "nachzurüsten". (mm)

Listings

```
*************************************************************
*              AMIGA DOS INFO                               *
*       (C)1990 by Peter Woop & AMIGA DOS                   *
*       Name             :Fraktalgenerator                  *
*       Sprache          :ASSEMBLER                         *
*       Besonderheiten   :keine                             *
*************************************************************

*---- library-offsets ---------------------------------*

execbase = 4

allocmem = -198
freemem = -210
getmsg = -372
closelibrary = -414
openlibrary = -552

closescreen = -66
closewindow = -72
displaybeep = -96
openscreen = -198
openwindow = -204
refreshgadgets = -222
screentofront = -252
wbenchtofront = -342
remakedisplay = -384

clearscreen = -48
move = -240
draw = -246
writepixel = -324
setapen = -342
setdrmd = -354

open = -30
close = -36
write = -48

*---- programmstart ------------------------------------*

beginning:

   move.l   execbase,a6

*---- benoetigte libraries oeffnen ------------------*

   lea      dosname,a1          ; dos-library
   moveq    #0,d0
   jsr      openlibrary(a6)
   move.l   d0,dosbase          ; zeiger sichern
   beq      no_dos              ; bei fehler programmende

   lea      intname,a1          ; intuition-library
   moveq    #0,d0
   jsr      openlibrary(a6)
   move.l   d0,intbase
   beq      no_int

   lea      gfxbase,a1          ; graphics-library
   moveq    #0,d0
   jsr      openlibrary(a6)
   move.l   d0,gfxbase
   beq      no_gfx

*---- screen und window oeffnen --------------------*


   move.l   intbase,a6
   lea      screen_defs,a0
   jsr      openscreen(a6)
   move.l   d0,screen_hd
   beq      no_screen

   lea      window_defs,a0
   jsr      openwindow(a6)
   move.l   d0,window_hd
   beq      no_window

*---- view- und rastport ermitteln -----------------*

   move.l   screen_hd,a0
   add.l    #44,a0
   move.l   a0,viewport
   add.l    #40,a0
   move.l   a0,rastport

*---- bitplane-struktur umbauen --------------------*

   add.l    #108,a0             ; zeiger auf erste plane
   moveq    #4,d7               ; 5 planes sind vorhanden

free_plane_memory:

   move.l   execbase,a6         ; speicher der einzelnen
   move.l   (a0)+,a1            ; planes freigeben
   move.l   #10240,d0
   move.l   a0,-(sp)
   jsr      freemem(a6)
   move.l   (sp)+,a0
   dbra     d7,free_plane_memory

   move.l   #51200,d0                 ; zusammenhaengenden
   move.l   #$10002,d1                ; speicher fuer planes
   jsr      allocmem(a6)              ; holen

   move.l   screen_hd,a0
   add.l    #192,a0
   moveq    #4,d7

set_new_pointers:

   move.l   d0,(a0)+                  ; adressen in bitplane-
   add.l    #10240,d0                 ; struktur eintragen
   dbra     d7,set_new_pointers

   move.l   intbase,a6                ; bildschirm neu aufbauen
   jsr      remakedisplay(a6)

*---- kommando-window oeffnen ----------------------*

   jsr      wbenchtofront(a6)

   lea      wbwind_defs,a0            ; fenster oeffnen
   jsr      openwindow(a6)
   move.l   d0,wbwind_hd              ; alles ok
   beq      no_wbwind                 ; nein => fehler

   clr      pic_on_screen             ; flagge runter
   bsr      bin_to_asc                ; koordinaten nach ascii

wait_for_message:


   move.l   execbase,a6
   move.l   wbwind_hd,a0
   move.l   86(a0),a0
   jsr      getmsg(a6)                ; message holen
   tst.l    d0                        ; war was ?
   beq.s    wait_for_message          ; nein => weiter warten
   move.l   d0,a0
   move     22(a0),d1                 ; ereignis nach d1
   cmp      #$40,d1                   ; gadget angeklickt ?
   beq.s    gadget_up                 ; ja => runter
   cmp      #$200,d1                  ; close-gadget ?
   bne.s    wait_for_message          ; nein => wieder rauf

*---- screen und windows wieder schliessen ----------*

   move.l   intbase,a6
   move.l   wbwind_hd,a0
   jsr      closewindow(a6)

no_wbwind:

   move.l   window_hd,a0
   jsr      closewindow(a6)

no_window:

   move.l   screen_hd,a0
   jsr      closescreen(a6)

*---- libraries schliessen -------------------------*

no_screen:

   move.l   execbase,a6
   move.l   gfxbase,a1
   jsr      closelibrary(a6)

no_gfx:

   move.l   intbase,a1
   jsr      closelibrary(a6)

no_int:

   move.l   dosbase,a1
   jsr      closelibrary(a6)

*---- programmende ---------------------------------*

no_dos:

   rts

*---- gadget-kommandos ausfuehren ------------------*

gadget_up:

   move.l   28(a0),a0
   move     38(a0),d1
   cmp      #7,d1                     ; fraktal berechnen ?
   beq      make_fractal              ; ja => dann los
   cmp      #8,d1                     ; ausschnitt bestimmen ?
   beq      new_part                  ; ja => auf gehts
```

Listing: Fraktalgenerator

```
   cmp     #9,d1                 ; polieren ?
   beq     smooth
   cmp     #10,d1

   beq     save_picture          ; bild speichern
   bra     wait_for_message      ; wieder in die schleife

*---- apfelmaennchen berechnen ----------------------*

make_fractal:

   move    gdinfo_05+30,max_iter
   bsr     check_entry           ; eingaben pruefen
   bsr     asc_to_bin            ; koordinaten wandeln
   bsr     fractal_to_front      ; bildschirm nach vorn
   move.l  screen_hd,a2          ; a2: zeiger erste plane
   move.l  192(a2),a2
   move.l  a2,-(sp)              ; a2 auf stack sichern
   move    #12799,d7             ; bildschirm loeschen

clear_screen:

   clr.l   (a2)+                 ; 12800 lang-worte
   dbra    d7,clear_screen

   move.l  (sp)+,a2              ; plane-zeiger reparieren
   bsr     make_table            ; koordinatentab. aufbauen

   lea     table_q,a6            ; a6: zeiger ver.-tabelle

   move    #255,d1               ; screen hat 256 zeilen

y_loop:

   move    d1,-(sp)              ; vertikal-zaehler sichern
   lea     table_p,a5            ; a5: zeiger hor.-tabelle
   bsr     iteration             ; iteration durchfuehren
   move    d0,store_00           ; ergebnis speichern

   move    (sp)+,d1
   move    #19,d0

x_loop:

   movem   d0/d1,-(sp)
   add.l   #32,a5
   bsr     iteration
   move    d0,store_08
   cmp     store_00,d0
   bne.s   erstes_byte_kaputt
   move    #$ff00,d1
   bsr     plot
   bra     erstes_byte_okay

erstes_byte_kaputt:

   sub.l   #16,a5
   bsr     iteration
   move    d0,store_04
   cmp     store_00,d0
   bne.s   erstes_nibble_kaputt
   move    #$f000,d1
   bsr     plot
   bra     erstes_nibble_okay

erstes_nibble_kaputt:

   move    store_00,d0
   move    #$8000,d1
   bsr     plot

   sub.l   #12,a5
   bsr     iteration
   move    #$4000,d1
   bsr     plot
   addq.l  #4,a5
   bsr     iteration
   move    #$2000,d1
   bsr     plot
   addq.l  #4,a5
   bsr     iteration
   move    #$1000,d1
   bsr     plot
   addq.l  #4,a5
   move    store_04,d0

erstes_nibble_okay:

   cmp     store_08,d0
   bne.s   zweites_nibble_kaputt
   move    #$0f00,d1
   bsr     plot
   add.l   #16,a5
   bra     erstes_byte_okay

zweites_nibble_kaputt:

   move    store_04,d0
   move    #$0800,d1
   bsr     plot
   addq.l  #4,a5
   bsr     iteration
   move    #$0400,d1
   bsr     plot
   addq.l  #4,a5
   bsr     iteration
   move    #$0200,d1
   bsr     plot
   addq.l  #4,a5
   bsr     iteration
   move    #$0100,d1
   bsr     plot
   addq.l  #4,a5

erstes_byte_okay:

   add.l   #32,a5
   bsr     iteration
   move    d0,store_10
   cmp     store_08,d0
   bne.s   zweites_byte_kaputt
   move    #$00ff,d1
   bsr     plot
   bra     zweites_byte_okay
```

Listing: Fraktalgenerator

```
zweites_byte_kaputt:

   sub.l    #16,a5
   bsr      iteration
   move     d0,store_0c
   cmp      store_08,d0
   bne.s    drittes_nibble_kaputt
   move     #$00f0,d1
   bsr      plot
   bra      drittes_nibble_okay

drittes_nibble_kaputt:

   move     store_08,d0
   move     #$0080,d1
   bsr      plot
   sub.l    #12,a5
   bsr      iteration
   move     #$0040,d1
   bsr      plot
   addq.l   #4,a5
   bsr      iteration
   move     #$0020,d1
   bsr      plot
   addq.l   #4,a5
   bsr      iteration
   move     #$0010,d1
   bsr      plot
   addq.l   #4,a5

drittes_nibble_okay:

   move     store_0c,d0
   cmp      store_10,d0
   bne.s    viertes_nibble_kaputt
   move     #$000f,d1
   bsr      plot
   add.l    #16,a5
   bra      zweites_byte_okay

viertes_nibble_kaputt:

   move     store_0c,d0
   move     #$0008,d1
   bsr      plot
   addq.l   #4,a5
   bsr      iteration
   move     #$0004,d1
   bsr      plot
   addq.l   #4,a5
   bsr      iteration
   move     #$0002,d1
   bsr      plot
   addq.l   #4,a5
   bsr      iteration
   move     #$0001,d1
   bsr      plot
   addq.l   #4,a5

zweites_byte_okay:

   move     store_10,store_00
   addq.l   #2,a2
   movem    (sp)+,d0/d1
   dbra     d0,x_loop
   btst     #6,$bfe001
   beq.s    exit_fractal
   addq.l   #4,a6
   dbra     d1,y_loop

exit_fractal:

   move     #-1,pic_on_screen
   bsr      wb_to_front
   bra      wait_for_message

*---- tabelle mit koordinatenwerten aufbauen --------*

make_table:

   move.l   realfr,d0           ; differenz der anfangs-
   move.l   realto,d1           ; koordinaten bilden
   sub.l    d0,d1
   move     #320,d7             ; durch 320 teilen
   bsr      divide

   lea      table_p,a0
   move     #319,d7             ; 320 werte eintragen

p_loop:

   move.l   d0,(a0)+            ; koordinaten jedes bild-
   add.l    d1,d0               ; schirmpunktes in tabel.
   dbra     d7,p_loop

   move.l   imagfr,d0           ; abstand der imaginaeren
   move.l   imagto,d1           ; grenzen bestimmen
   sub.l    d0,d1
   move     #256,d7             ; durch 256 teilen
   bsr      divide

   lea      table_q,a0
   move     #255,d7             ; 256 werte

q_loop:

   move.l   d0,(a0)+            ; jetzt koordinaten von
   add.l    d1,d0               ; zeilen eintragen
   dbra     d7,q_loop
   rts

*---- division (32 bit) ----------------------------*

divide:

   move.l   d1,d2
   clr      d1
   swap     d1
   divu     d7,d1
   move.l   d1,d3
   swap     d1
   move     d2,d3
   divu     d7,d3
   move     d3,d1
   rts

*---- iterationsroutine ----------------------------*

iteration:

   sub.l    a0,a0               ; x
   sub.l    a1,a1               ; y
   moveq    #0,d5               ; zaehler f. iterationen

iteration_loop:

   move.l   a1,d0               ; ist y negativ ?
   btst     #31,d0
   beq.s    y_positive
   neg.l    d0                  ; => vorzeichen umdrehen

y_positive:

   move.l   d0,d1

   move     d1,d2
   swap     d1
   mulu     d1,d2
   moveq    #0,d3
   lsl.l    #1,d2
   roxl.l   #1,d3
   mulu     d1,d1
   moveq    #0,d0
   move     d2,d0
   swap     d0
   clr      d2
   swap     d2
   add.l    d3,d2
   add.l    d2,d1
   move.l   d1,d2
   swap     d2
   and      #$f800,d2
   bne      overflow
   rol.l    #4,d1
   and.b    #$f0,d1
   rol.l    #4,d0
   and.b    #$0f,d0
   or.b     d0,d1
   move.l   d1,d6

   move.l   a0,d0               ; ist x negativ ?
   btst     #31,d0
   beq.s    x_positive
   neg.l    d0                  ; => vorzeichen umdrehen

x_positive:

   move.l   d0,d1
   move     d1,d2
   swap     d1
   mulu     d1,d2
   moveq    #0,d3
   lsl.l    #1,d2
   roxl.l   #1,d3
   mulu     d1,d1
   moveq    #0,d0
   move     d2,d0
   swap     d0
   clr      d2
   swap     d2
   add.l    d3,d2
   add.l    d2,d1
   move.l   d1,d2
   swap     d2
   and      #$f800,d2
   bne      overflow
   rol.l    #4,d1
   and.b    #$f0,d1
   rol.l    #4,d0
   and.b    #$0f,d0
   or.b     d0,d1
   move.l   d1,d7

   move.l   d6,d0               ; xx+yy>8 ?
   add.l    d7,d0
   bcs      overflow            ; => abbruch
   bmi      overflow

   move.l   a1,d0               ; y nach d0
   move.l   a0,d1               ; x nach d1
   clr      d4                  ; vorzeichenflag loeschen

   move.l   d0,d2               ; y zusaetzlich nach d2
   btst     #31,d2              ; ist y negativ ?
   beq.s    y_not_negative
   neg.l    d2                  ; => d0 und d2 negieren
   neg.l    d0
   addq     #1,d4               ; vorzeichenflag plus 1

y_not_negative:

   move.l   d1,d3               ; x zusaetzlich nach d3
   btst     #31,d3              ; ist x negativ ?
   beq.s    x_not_negative
   neg.l    d3                  ; => d1 und d3 negieren
   neg.l    d1
   addq     #1,d4               ; vorzeichenflag plus 1

x_not_negative:

   move     d4,prefix           ; vorzeichen merken

   move.l   d0,d2
   swap     d2
   mulu     d1,d2
   move.l   d1,d3
   swap     d3
   mulu     d0,d3
   swap     d0
   swap     d1
   mulu     d1,d0
   moveq    #0,d4
   move     d2,d4
   swap     d4
```

Listing: Fraktalgenerator

```
clr      d2
swap     d2
moveq    #0,d1
move     d3,d1
swap     d1
clr      d3
swap     d3
add.l    d4,d1
addx.l   d2,d3
add.l    d3,d0
rol.l    #5,d0
and.b    #$e0,d0
rol.l    #5,d1
and.b    #$1f,d1
or.b     d1,d0
move     prefix,d4          ; bit 0 im vz-flag ?
btst     #0,d4              ; => x oder y war negativ
beq.s    result_positive
neg.l    d0                 ; also ergebnis negieren

result_positive:

move.l   d0,a0
sub.l    (a6),d0
move.l   d0,a0
move.l   d6,d1
move.l   d7,d0
sub.l    d0,d1
sub.l    (a5),d1
move.l   d1,a1
addq     #1,d5              ; iterationszaehler plus 1
cmp      max_iter,d5        ; grenze erreicht ?
bne      iteration_loop     ; => ende, sonst nochmal
move     #0,d5              ; farbe 0 fuer grenze

overflow:

move     d5,d0              ; farbe nach d0
and      #15,d0             ; nur werte von 0 bis 31
rts

*---- plot-routine ----------------------------------*

plot:

asl      #4,d0              ; farbwert mal 16
jmp      zero(pc,d0.w)      ; in entsprechende routine

zero:

rts                         ; farbe 0 (hier nix tun)
blk.b    14,0

or       d1,0(a2)           ; 1
rts
blk.b    10,0

or       d1,10240(a2)       ; 2
rts
blk.b    10,0

or       d1,0(a2)           ; 3
or       d1,10240(a2)
rts
blk.b    6,0

or       d1,20480(a2)       ; 4
rts
blk.b    10,0

or       d1,0(a2)           ; 5
or       d1,20480(a2)
rts
blk.b    6,0

or       d1,10240(a2)       ; 6
or       d1,20480(a2)
rts
blk.b    6,0

or       d1,0(a2)           ; 7
or       d1,10240(a2)
or       d1,20480(a2)
rts
blk.b    2,0

or       d1,30720(a2)       ; 8
rts
blk.b    10,0

or       d1,0(a2)           ; 9
or       d1,30720(a2)
rts
blk.b    6,0

or       d1,10240(a2)       ; 10
or       d1,30720(a2)
rts
blk.b    6,0

or       d1,0(a2)           ; 11
or       d1,10240(a2)
or       d1,30720(a2)
rts
blk.b    2,0

or       d1,20480(a2)       ; 12
or       d1,30720(a2)
rts
blk.b    6,0

or       d1,0(a2)           ; 13
or       d1,20480(a2)
or       d1,30720(a2)
rts
blk.b    2,0

or       d1,10240(a2)       ; 14
or       d1,20480(a2)
or       d1,30720(a2)
rts
blk.b    2,0
```

Listing: Fraktalgenerator

```

or       d1,0(a2)           ; 15
or       d1,10240(a2)
or       d1,20480(a2)
or       d1,30720(a2)
rts

*---- bildschirmteile polieren ----------------------*

smooth:

tst      pic_on_screen      ; nur wenn bild berechnet
bne.s    smoothing_possible

bsr      flash_workbench    ; sonst screen flashen
beq      wait_for_message

smoothing_possible:

bsr      get_part           ; ausschnitt bestimmen
bsr      erase_cross        ; fadenkreuz loeschen

move.l   gfxbase,a6         ; zeichenmodus setzen
move.l   rastport,a1
moveq    #0,d0
jsr      setdrmd(a6)

bsr      mousekey_hold

lea      table_p,a5         ; ausgeaehlten koordinaten
moveq    #0,d0              ; entsprechende adresse
move     x1,d0              ; in tabelle ermitteln
lsl.l    #2,d0
add.l    d0,a5
move.l   a5,p_pointer       ; sichern
lea      table_q,a6
moveq    #0,d0
move     y1,d0
lsl.l    #2,d0
add.l    d0,a6

move     y1,d1              ; erste zu rechnende zeile

y_count:

move.l   p_pointer,a5
move     x1,d0              ; erste spalte

x_count:

movem.l  d0/d1/a5/a6,-(sp)  ; auf den stack
movem    d0/d1,-(sp)        ; dito
jsr      iteration          ; iteration durchfuehren
move.l   gfxbase,a6
move.l   rastport,a1
jsr      setapen(a6)        ; farbe setzen
movem    (sp)+,d0/d1
jsr      writepixel(a6)     ; und dann den punkt
movem.l  (sp)+,d0/d1/a5/a6
addq.l   #4,a5              ; x-zeiger erhoehen
addq     #1,d0
cmp      x2,d0              ; ende der zeile ?
ble.s    x_count            ; nein => weitermachen
btst     #6,$bfe001         ; bei maustaste abbrechen
beq.s    exit_smoothing
addq.l   #4,a6              ; y-zeiger auch plus 1
addq     #1,d1              ; naechste zeile
cmp      y2,d1              ; alles fertig ?
ble.s    y_count            ; nein => schleife

exit_smoothing:

bsr      wb_to_front
bra      wait_for_message

*---- neuen ausschnitt bestimmen --------------------*

new_part:

bsr      get_part           ; Ausschnitt waehlen
bsr      make_coordinates   ; Koordinates bestimmen
bsr      bin_to_asc         ; festkomma in ascii
bsr      wb_to_front        ; workbench nach vorn
bsr      erase_cross        ; fadenkreuze loeschen
bra      wait_for_message

make_coordinates:

lea      table_p,a0         ; koordinate des
move     x1,d0              ; punktes neuer anfang
lsl      #2,d0
move.l   (a0,d0.w),realfr   ; real-anfang

move     x2,d0
lsl      #2,d0
move.l   (a0,d0.w),realto   ; real-ende

lea      table_q,a0
move     y1,d0
lsl      #2,d0
move.l   (a0,d0.w),imagfr   ; imaginaer-anfang

move     y2,d0
lsl      #2,d0
move.l   (a0,d0.w),imagto   ; imaginaer-ende

rts

*---- ausschnitt waehlen ----------------------------*

get_part:

tst      pic_on_screen      ; schon bild gerechnet ?
bne.s    picture_there      ; ja => also weitermachen

bsr      flash_workbench    ; bildschirm aufblitzen
addq.l   #4,sp              ; stackpointer korrig.
bra      wait_for_message   ; heim ins menue

picture_there:

move.l   gfxbase,a6
```

Listing: Fraktalgenerator

```
        move.l  rastport,a1
        moveq   #2,d0
        jsr     setdrmd(a6)

        bsr     fractal_to_front     ; screen nach vorn holen
        move.l  window_hd,a5         ; zeiger auf das fenster
        bsr     part_sub             ; koordinaten links oben
        move    d6,y1                ; bestimmen
        move    d7,x1

        bsr     mousekey_hold        ; warten, bis taste weg

        bsr     part_sub             ; koordinaten rechts u.
        move    d6,y2                ; bestimmen
        move    d7,x2

        move    x1,d0                ; anfangskoordinate klei-
        cmp     x2,d0                ; ner endkoordinate ?
        blt.s   x1_lt_x2

        move    x2,x1                ; sonst rumdrehen
        move    d0,x2

x1_lt_x2:

        move    y1,d0                ; dito fuer y-werte
        cmp     y2,d0
        blt.s   y1_lt_y2

        move    y2,y1
        move    d0,y2

y1_lt_y2:

        rts

part_sub:

        move    12(a5),d6            ; maus y-koordinate
        move    14(a5),d7            ; x
        bsr     draw_cross           ; fadenkreuz zeichnen

mouse_not_moved:

        cmp     14(a5),d7            ; koordinaten noch gleich
        bne.s   mouse_moved          ; => maus nicht bewegt
        cmp     12(a5),d6
        bne.s   mouse_moved
        btst    #6,$bfe001           ; linke maustaste
        bne.s   mouse_not_moved      ; => schleife beenden

        bsr     draw_cross           ; altes kreuz loeschen
        move    12(a5),d6            ; aktuelle koordinaten
        move    14(a5),d7            ; sichern

        bra     draw_cross           ; fadenkreuz malen

mouse_moved:

        bsr     draw_cross           ; kreuz loeschen
        bra.s   part_sub             ; wieder in die schleife

*---- fadenkreuz zeichnen ---------------------------*

draw_cross:

        move.l  gfxbase,a6           ; horizontale linie
        move.l  rastport,a1
        moveq   #0,d0
        move    d6,d1
        jsr     move(a6)
        move    #319,d0
        move    d6,d1
        jsr     draw(a6)

        move    d7,d0                ; jetzt die vertikale
        moveq   #0,d1
        jsr     move(a6)
        move    d7,d0
        move    #255,d1
        jmp     draw(a6)

erase_cross:

        move    x1,d7
        move    y1,d6
        bsr.s   draw_cross
        move    x2,d7
        move    y2,d6
        bra.s   draw_cross

*---- eingaben ueberpruefen -------------------------*

check_entry:

        rts

        lea     gdbuff_01,a0
        bsr     ce_sub

        lea     gdbuff_02,a0
        bsr     ce_sub

        lea     gdbuff_03,a0
        bsr     ce_sub

        lea     gdbuff_04,a0

ce_sub:

        move.b  (a0)+,d0             ; vorzeichen nach d0
        cmp.b   #"-",d0              ; minus ?
        beq.s   prefix_ok            ; ja => alles klar
        cmp.b   #"+",d0              ; plus ?
        beq.s   prefix_ok            ; ja => auch in ordnung

        moveq   #9,d7                ; sonst alles 1 nach
        bsr     move_it              ; hinten
        move.b  #"+",-1(a0)          ; plus einsetzen

prefix_ok:

        cmp.b   #".",(a0)            ; zeichen 2 nicht punkt ?
```

Listing: Fraktalgenerator

```
        bne.s   digit_found          ; ja => dann weiter

        moveq   #8,d7                ; sonst auch hier zurueck
        bsr     move_it
        move.b  #"0",(a0)            ; null an zweite stelle

digit_found:

        move.l  intbase,a6           ; gadgets aktualisieren
        lea     gadget_01,a0
        move.l  wbwind_hd,a1
        sub.l   a2,a2
        jmp     refreshgadgets(a6)

move_it:

        move.l  a0,a1
        addq.l  #8,a1

move_loop:

        move.b  (a1),1(a1)
        subq.l  #1,a1
        dbra    d7,move_loop
        clr.b   9(a0)
        rts

*---- koordinaten von binaer nach ascii wandeln -----*

bin_to_asc:

        move.l  realfr,d0
        lea     gdbuff_01,a0
        bsr     ba_sub

        move.l  realto,d0
        lea     gdbuff_02,a0
        bsr     ba_sub

        move.l  imagfr,d0
        lea     gdbuff_03,a0
        bsr     ba_sub

        move.l  imagto,d0
        lea     gdbuff_04,a0
        bsr     ba_sub

        move.l  intbase,a6
        lea     gadget_01,a0
        move.l  wbwind_hd,a1
        sub.l   a2,a2
        jmp     refreshgadgets(a6)

ba_sub:

        moveq   #8,d7
        move.b  #"+",(a0)            ; plus an anfang setzen
        tst.l   d0                   ; koordinatenwert testen
        bge.s   positive             ; positiv => weiter
        move.b  #"-",(a0)            ; sonst minus einsetzen
        neg.l   d0                   ; und wert negieren

positive:

        addq.l  #1,a0                ; naechste stelle
        lea     table,a1             ; ptr auf wertigkeiten

ba_loop_01:

        move.l  (a1)+,d2             ; wertigkeit aus tabelle
        move.b  #48,d6               ; ascii "0" nach d6

ba_loop_02:

        sub.l   d2,d0                ; wertigkeit abziehen
        bmi.s   ba_exit              ; erg. negativ => weiter
        addq.b  #1,d6                ; ascii-wert plus eins
        cmp.b   #57,d6               ; "9" schon erreicht ?
        bne.s   ba_loop_02           ; nein => weiter
        sub.l   d2,d0

ba_exit:

        move.b  d6,(a0)+             ; ascii-wert eintragen
        add.l   d2,d0                ; wert wieder addieren
        cmp     #8,d7                ; stelle acht bearbeitet?
        bne.s   no_point             ; ja => punkt setzen
        move.b  #".",(a0)+

no_point:

        dbra    d7,ba_loop_01        ; wieder in die schleife
        clr.b   -(a0)                ; letzte stelle auf null
        cmp.b   #"9",-(a0)           ; letzte stelle eine 9 ?
        bne.s   no_round             ; nein => nicht runden

round_loop:

        move.b  #"0",(a0)            ; sonst gegen 0 tauschen

point:

        cmp.b   #"9",-(a0)           ; naechste stelle auch 9?
        beq.s   round_loop           ; ja => weiter runden
        cmp.b   #".",(a0)            ; punkt ueberspringen
        beq.s   point
        addq.b  #1,(a0)

no_round:

        rts

*---- jetzt in der umgekehrten richtung wandeln -----*

asc_to_bin:

        lea     gdbuff_01,a0
        bsr     ab_sub
        move.l  d0,realfr

        lea     gdbuff_02,a0
        bsr     ab_sub
        move.l  d0,realto

```

Listing: Fraktalgenerator

```
  lea       gdbuff_03,a0
  bsr       ab_sub
  move.l    d0,imagfr
  lea       gdbuff_04,a0
  bsr       ab_sub
  move.l    d0,imagto
  rts
ab_sub:
  move.b    (a0)+,d5              ; vorzeichen merken
  cmp.b     #48,d5                ; zeichen < asc "0" ?
  blt.s     first_no_digit        ; => wird vorzeichen sein
  move.b    #"+",d5               ; sonst plus annehmen
  subq.l    #1,a0                 ; pointer minus 1
first_no_digit:
  lea       table,a1              ; ptr auf wertetab.
  moveq     #0,d0                 ; zahl erstmal auf null
  moveq     #8,d7
ab_loop_01:
  move.l    (a1)+,d2              ; wert aus tabelle holen
  moveq     #0,d6
  move.b    (a0)+,d6              ; eine stelle nach d6
  sub       #48,d6                ; 48 (asc "0") abziehen
  beq.s     digit_is_zero         ; ergebnis <= 0 ?
  bmi.s     digit_is_zero         ; => nichts tun
ab_loop_02:
  add.l     d2,d0                ; wert der stelle addieren
  subq      #1,d6                 ; das ganze d6 mal
  bne.s     ab_loop_02 ;bgt ?
digit_is_zero:
  cmp       #8,d7                 ; stelle 8 an der reihe ?
  bne.s     digit_ok              ; nein => dann alles ok
  move.b    (a0)+,d6              ; naechste stelle lesen
digit_ok:
  dbra      d7,ab_loop_01         ; naechste stelle
  cmp.b     #"+",d5               ; vorzeichen positiv ?
  beq.s     value_is_positive     ; => wert so lassen
  neg.l     d0                    ; ansonsten negieren
value_is_positive:
  rts
*---- screens umschalten ----------------------------*
wb_to_front:
  move.l    intbase,a6
  jmp       wbenchtofront(a6)
fractal_to_front:
  move.l    intbase,a6
  move.l    screen_hd,a0
  jmp       screentofront(a6)
*---- workbench aufblitzen lassen -------------------*
flash_workbench:
  move.l    intbase,a6
  sub.l     a0,a0
  jmp       displaybeep(a6)
*---- warten, bis linke maustaste losgelassen -------*
mousekey_hold:
  btst      #6,$bfe001
  beq.s     mousekey_hold
  rts
*---- bild auf diskette abspeichern -----------------*
save_picture:
  tst.b     gdbuff_06
  bne.s     filename_there
  bsr       flash_workbench
  bra       wait_for_message
filename_there:
  move.l    dosbase,a6
  move.l    #gdbuff_06,d1
  move.l    #1006,d2
  jsr       open(a6)
  move.l    d0,file_hd
  beq.s     file_error
  move.l    #iff_header,d2
  move.l    #48,d3
  bsr       write_data
  move.l    screen_hd,a4
  add.l     #192,a4
  moveq     #0,d4      ; zeile
loop_y:
  moveq     #0,d5      ; plane
loop_p:
  move.l    (a4,d5.w),d2
  add.l     d4,d2
  moveq     #40,d3
  bsr       write_data
```

Listing: Fraktalgenerator

```
  addq      #4,d5
  cmp       #16,d5
  blt.s     loop_p
  add.l     #40,d4
  cmp.l     #10240,d4
  blt.s     loop_y
close_file:
  move.l    dosbase,a6
  move.l    file_hd,d1
  jsr       close(a6)
  beq       file_error
  bra       wait_for_message
write_data:
  move.l    file_hd,d1
  jsr       write(a6)
  cmp.l     #-1,d0
  bne.s     write_okay
  addq.l    #4,sp
  bsr       flash_workbench
  bra.s     close_file
write_okay:
  rts
file_error:
  bsr       flash_workbench
  bra       wait_for_message
iff_header:
  dc.b      "FORM"
  dc.l      41000
  dc.b      "ILBMBMHD"
  dc.l      20
  dc.w      320,256,0,0
  dc.b      4,0,0,0
  dc.w      0
  dc.b      10,11
  dc.w      320,256
  dc.b      "BODY"
  dc.l      40960
file_hd:
  dc.l      0
*---- gadget-strukturen ------------------------------*
gadget_01:      dc.l    gadget_02
                dc.w    150,20,136,10,0,3,4
                dc.l    gdbord_01,0,gdtext_01,0,gdinfo_01
                dc.w    1,0,0
gdinfo_01:      dc.l    gdbuff_01,0
                dc.w    0,11,0,0,0,0,0,0
                dc.l    0,0,0
gdbuff_01:      blk.b   16,0
gdtext_01:      dc.b    1,0,1,0
                dc.w    -110,0
                dc.l    0,gdtptr_01,0
gdtptr_01:      dc.b    "Realteil von",0,0
gdbord_01:      dc.w    0,0
                dc.b    1,1,0,5
                dc.l    gdkord_01,0
gdkord_01:      dc.w    -3,-3,137,-3,137,10,-3,10,-3,-3
gadget_02:      dc.l    gadget_03
                dc.w    340,20,136,10,0,3,4
                dc.l    gdbord_01,0,gdtext_02,0,gdinfo_02
                dc.w    2,0,0
gdinfo_02:      dc.l    gdbuff_02,0
                dc.w    0,11,0,0,0,0,0,0
                dc.l    0,0,0
gdbuff_02:      blk.b   16,0
gdtext_02:      dc.b    1,0,1,0
                dc.w    -38,0
                dc.l    0,gdtptr_02,0
gdtptr_02:      dc.b    "bis",0
gadget_03:      dc.l    gadget_04
                dc.w    150,36,136,10,0,3,4
                dc.l    gdbord_01,0,gdtext_03,0,gdinfo_03
                dc.w    3,0,0
gdinfo_03:      dc.l    gdbuff_03,0
                dc.w    0,11,0,0,0,0,0,0
                dc.l    0,0,0
gdbuff_03:      blk.b   16,0
gdtext_03:      dc.b    1,0,1,0
                dc.w    -110,0
                dc.l    0,gdtptr_03,0
gdtptr_03:      dc.b    "Imagteil von",0,0
gadget_04:      dc.l    gadget_05
                dc.w    340,36,136,10,0,3,4
                dc.l    gdbord_01,0,gdtext_02,0,gdinfo_04
                dc.w    4,0,0
gdinfo_04:      dc.l    gdbuff_04,0
                dc.w    0,11,0,0,0,0,0,0
                dc.l    0,0,0
gdbuff_04:      blk.b   16,0
gadget_05:      dc.l    gadget_06
                dc.w    150,52,136,10,0,$800,4
                dc.l    gdbord_01,0,gdtext_05,0,gdinfo_05
                dc.w    5,0,0
gdinfo_05:      dc.l    gdbuff_05,0
                dc.w    0,5,0,0,0,0,0,0
                dc.l    0,20,0
gdbuff_05:      dc.b    "20",0,0,0,0,0,0
gdtext_05:      dc.b    1,0,1,0
                dc.w    -110,0
                dc.l    0,gdtptr_05,0
```

Listing: Fraktalgenerator

```
gdtptr_05:      dc.b    "Rechentiefe:",0,0

gadget_06:      dc.l    gadget_07
                dc.w    40,103,280,10,0,3,4
                dc.l    gdbord_06,0,gdtext_06,0,gdinfo_06
                dc.w    6,0,0
gdinfo_06:      dc.l    gdbuff_06,0
                dc.w    0,35,0,0,0,0,0,0
                dc.l    0,0,0
gdbuff_06:      blk.b   40,0
gdtext_06:      dc.b    1,0,1,0
                dc.w    6,-13
                dc.l    0,gdtptr_06,0
gdtptr_06:      dc.b    "Filename:",0
gdbord_06:      dc.w    0,0
                dc.b    1,1,0,5
                dc.l    gdkord_06,0
gdkord_06:      dc.w    -3,-3,284,-3,284,10,-3,10,-3,-3

gadget_07:      dc.l    gadget_08
                dc.w    340,56,136,8,0,3,1
                dc.l    gdbord_01,0,gdtext_07,0,0
                dc.w    7,0,0
gdtext_07:      dc.b    1,0,1,0
                dc.w    12,0
                dc.l    0,gdtptr_07,0

gdtptr_07:      dc.b    "Bild berechnen",0,0

gadget_08:      dc.l    gadget_09
                dc.w    340,71,136,8,0,3,1
                dc.l    gdbord_01,0,gdtext_08,0,0
                dc.w    8,0,0
gdtext_08:      dc.b    1,0,1,0
                dc.w    28,0
                dc.l    0,gdtptr_08,0
gdtptr_08:      dc.b    "Ausschnitt",0,0

gadget_09:      dc.l    gadget_10
                dc.w    340,87,136,8,0,3,1
                dc.l    gdbord_01,0,gdtext_09,0,0
                dc.w    9,0,0
gdtext_09:      dc.b    1,0,1,0
                dc.w    36,0
                dc.l    0,gdtptr_09,0
gdtptr_09:      dc.b    "Polieren",0,0

gadget_10:      dc.l    0
                dc.w    340,103,136,8,0,3,1
                dc.l    gdbord_01,0,gdtext_10,0,0
                dc.w    10,0,0
gdtext_10:      dc.b    1,0,1,0
                dc.w    12,0
                dc.l    0,gdtptr_10,0
gdtptr_10:      dc.b    "Bild speichern",0,0

*---- variablen und definitionen --------------------*

dosname:        dc.b    "dos.library",0
dosbase:        dc.l    0

intname:        dc.b    "intuition.library",0
intbase:        dc.l    0

gfxbase:        dc.b    "graphics.library",0,0
gfxname:        dc.l    0

viewport:       dc.l    0
rastport:       dc.l    0

screen_defs:    dc.w    0,0,320,256,5
                dc.b    0,1
                dc.w    2,5
                dc.l    0,0,0,0

window_defs:    dc.w    0,0,320,256
                dc.b    0,1
                dc.l    $400,$11800,0,0,0
screen_hd:      dc.l    0,0
                dc.w    320,256,320,256,15
window_hd:      dc.l    0

wbwind_defs:    dc.w    40,40,560,120
                dc.b    0,1
                dc.l    $240,$1100e,gadget_01,0
                dc.l    wbwind_name,0,0
                dc.w    560,120,560,120,1
wbwind_name:    dc.b    "FrakGen AmigaDOS V1.01",0,0
wbwind_hd:      dc.l    0

realfr:         dc.l    $c8000000
realto:         dc.l    $37ffffff
imagfr:         dc.l    $c8000000

imagto:         dc.l    $37ffffff
max_iter:       dc.w    0

pic_on_screen:  dc.w    0
prefix:         dc.w    0

table_p:        blk.b   1280,0
table_q:        blk.b   1024,0
p_pointer:      dc.l    0

x1:             dc.w    0
y1:             dc.w    0
x2:             dc.w    0
y2:             dc.w    0

store_00:       dc.w    0
store_04:       dc.w    0
store_08:       dc.w    0
store_0c:       dc.w    0
store_10:       dc.w    0

table:          dc.l    $10000000/1
                dc.l    $10000000/10
                dc.l    $10000000/100
                dc.l    $10000000/1000
                dc.l    $10000000/10000
                dc.l    $10000000/100000
                dc.l    $10000000/1000000
                dc.l    $10000000/10000000
                dc.l    $10000000/100000000

*--------------------------------------------------------*
```

Listing: Fraktalgenerator



Gewußt wie!

Die Seiten für Einsteiger und Fortgeschrittene

Ein neuer Monat, eine neue AMIGA DOS und damit auch neue 'Gewußt wie'-Seiten. Zuerst jedoch mal ein Lob an alle Einsender: Wir finden es prima, daß sehr viele Einsender unserem Aufruf gefolgt sind und uns ihr Wissen zur Weitergabe an die Leser übermittelt haben. Deshalb Danke an alle, die bisher mitgemacht haben und die noch mitmachen werden.
Doch nun wieder zu unseren Kurztips.

HardError-Disks weiternutzen

Ist es Ihnen auch schon einmal passiert, daß sich plötzlich ein HardError auf einer Ihrer Disketten befand? Bisher konnten Sie solche Disks getrost dem Mülleimer überlassen. Damit ist jetzt Schluß! Sie brauchen nur die Mountlist, die sich im "devs"-Ordner Ihrer Startdiskette befindet, entsprechend abzuändern.

Ein Beispiel: Sie haben auf einer Diskette mehrere HardErrors auf den Spuren 70-79 und Sie besitzen zwei Diskettenlaufwerke. Fügen Sie folgenden Eintrag in die Mountlist Ihrer Startdiskette ein (die Zeilennummern nicht mit eingeben!). Dafür begeben Sie sich ins CLI oder in die Shell und geben dort ein "ed devs:mountlist", gefolgt von einem Return. Schreiben Sie den untenstehenden ersten Eintrag ans Ende der Mountlist, drücken Sie dann die Tasten <ESC> und <X>, gefolgt von einem Return.

Eintrag 1:

```
1  Krankenhaus: Device = trackdisk.device
2  Unit = 1
3  Flags = 1
4  Surfaces = 2
5  BlocksPerTrack = 11
6  Reserved = 2
7  Interleave = 0
8  LowCyl = 0 ; HighCyl = 69
9  Buffers = 20
10 BufMemType = 3
11 #
```

Der Wert LowCyl gibt die unterste zu formatierende Spur an (0), und HighCyl gibt die höchste zu formatierende Spur an (69). Normalerweise besitzt eine Disk die Spuren 0 bis 79, da aber 70 bis 79 defekt sind, können wir nur 0 bis 69 nutzen. Geben Sie nun im CLI/Shell den Befehl "mount Krankenhaus:" ein, um Ihr neues logisches Device Krankenhaus: anzumelden. Der Wert Unit in Zeile 2 gibt an, daß das Laufwerk df1: das Device Krankenhaus: simuliert. Sie können nun die defekte Diskette in Drive df1: einlegen und selbige mit folgendem Befehl formatieren:

sys:system/format drive Krankenhaus: name testdisk noicons

Die Diskette wird nun formatiert und ist für Sie wieder nutzbar, jedoch mit einer geringeren Speicherkapazität. Sollten Sie mehrere Disketten mit diesem Trick formatieren, die verschiedene defekte Spuren haben, so müssen Sie vor dem Einlegen einer anderen vorher defekten Disk das bisher genutze logische Device wieder abmelden (ist jedenfalls aus Speicherplatzgründen besser). Dies geht ganz einfach mit dem Befehl "assign remove Krankenhaus:". Nun können Sie das andere Device anmelden (z.B. mount Krankenhaus2:) und die Disk verwenden. Sie brauchen also für jede mit anderen Low- und HighCyl-Werten formatierte Diskette einen eigenen Mountlist-Eintrag. Für CLI-Geübte ist dies sicherlich einfach zu nutzen, alle anderen sollten damit erst einmal etwas herumprobieren (nur unformatierte, also leere Disketten benutzen).

FastFileSystem auf Disketten

Es wurde ja immer behauptet, man könne das FastFileSystem nur auf Festplatten und RamDisks anwenden. Stimmt aber nicht! Sie müssen nur folgenden Mountlist-Eintrag an die Mountlist Ihrer Startdiskette anfügen und schon kann's losgehen:

```
1  FFS-Drive: Device = trackdisk.device
2  FileSystem = l:FastFileSystem
3  Unit = 1
4  Flags = 1
5  Surfaces = 2
6  BlocksPerTrack = 11
7  Reserved = 2
8  Interleave = 0
9  LowCyl = 0 ; HighCyl = 79
10 Buffers = 20
11 GlobVec = -1
12 BufMemType = 3
13 Mount = 1
14 DosType = 0x444F5301
15 #
```

Melden Sie das neue logische Device mit "mount FFS-Drive:" an. Legen Sie nun eine leere, unformatierte Disk in Drive df1: und formatieren Sie diese mit folgendem Befehl:

sys:system/format drive FFS-Drive: name ffstestdisk ffs noicons

Die Diskette bietet nun 5% mehr Speicherplatz und wird bedeutend schneller geladen. Am extremsten bemerkt man dies beim "dir"-Befehl. Wollen Sie mehrere Disks mit dem FFS benutzen, so beachten Sie bitte, daß immer erst das FFS-Drive-Device mit "mount FFS-Drive:" vor dem ersten Einsatz einer FFS-formatierten Diskette angemeldet werden muß. Des weiteren beachten Sie bitte, daß das FastFileSystem keinen Diskettenwechsel erkennt, da eine Festplatte oder RamDisk im allgemeinen nicht gewechselt wird. Hierfür verwenden Sie bitte den "disk-

change"-Befehl, den Sie wie alle anderen Befehle im C-Verzeichnis Ihrer Diskette finden.

Schnelle Headerfiles mit dem 'Lattice C'-Compiler

Was Aztec-Besitzer schon lange genießen, ist nun auch Lattice-Usern seit der Version 5.02 ihres Compilers vergönnt. Ich spreche von der Möglichkeit, Headerfiles vorzucompilieren (auch PreCompiling). Hiermit müssen die Include-Files nicht immer neu übersetzt werden, sondern nur der Quelltext des Hauptmoduls. Hierbei wird das File sozusagen vorcompiliert. Die Includefiles werden als Symboltabelle im Quad:-Device gespeichert. Beim endgültigen Compilieren wird dieser Code wieder geladen. Die Include-Anweisungen im Quellprogramm müssen selbstverständlich wegfallen. Nehmen wir an, Ihr Programm benutzt nur die Headerfiles "dos.h" und "intuition/intuition.h". Um diese zu precompilen, erstellen Sie sich eine Datei mit dem Namen 'headers.c' (Beispielname!), die folgenden Inhalt hat:

#include <dos.h> #include <intuition/intuit ion.h>

Nun rufen Sie den Compiler mit folgender Anweisung auf:

lc1-ph headers.c

Er erzeugt eine Datei mit dem Namen 'headers.q'. Wollen Sie nun Ihr Hauptprogramm compilieren, so rufen Sie den lc1-Pass folgendermaßen auf:

lc1-Hheaders.q Hauptprogramm.c

Zwischen der Option -H und headers.q darf kein Leerzeichen sein! Im normalen Programmcode dürfen natürlich nicht mehr die Include-Dateien aufgerufen werden, die bereits übersetzt wurden. Andere Includes dürfen natürlich verwendet werden.

AutoRequest V1.1

"AReq" erzeugt einen AutoRequester im aktuellem Window mit den vom Benutzer übergebenen Parametern für Body, linkes und rechtes Gadget. Der Quellcode wurde mit dem 'Lattice C'-Compiler V5.02 erstellt, dürfte jedoch auch dem Aztec-C-V3.6a-Compiler keine Probleme bereiten. Es existieren weiter zwei Make-Dateien, die ich aufgrund ihrer Einfachheit als Batch-Dateien erstellt habe. 'make' erzeugt die normale Version des Programms und 'make.res' erzeugt die residentfähige Version. Weiterhin habe ich eine Beispiel-Batch-Datei beigefügt, die die Anwendung des Befehls demonstriert. Sollten noch Fragen offen sein, so stehe ich gerne über die Redaktion der AMIGA DOS zur Verfügung.

Übrigens, dies ist die Version 1.1 des Programmes. In dieser neuen Version habe ich die Argumentübergabe verändert und den Speicherbedarf des Programmes auf Diskette reduziert. Bei der normalen Version hat dies immerhin 192 Bytes eingebracht und bei der residentfähigen fast 165. Die größte Speicherplatzersparnis brachte die Entfernung der Abfrage, ob die 'Intuition.Library' geöffnet werden konnte. Diese Abfrage halte ich sowieso für überflüssig, da sie immer vorhanden ist. Es ist zwar kein guter Programmierstil, aber der Zweck heiligt die Mittel.

Info:
Programm: 'AReq'
Filelänge: 3276 Bytes
(Die Version für Speichersparer!)
Filelänge: 3360 Bytes
(Diese Version kann resident gemacht werden (RESIDENT-BEFEHL)!
Aufruf: 'AReq' Body LGadget RGadget
Body = Titel
RGadget = Text für rechtes Gadget
LGadget = Text für linkes Gadget

Texte können auch Leerzeichen beinhalten, müssen dann aber in Anführungszeichen übergeben werden.

Rückgabe:
FEHLER: 10 bei falschem Aufruf
Sonstige: 5 wenn linkes Gadget gedrückt (WARN)
0 wenn rechtes Gadget gedrückt
Beispiel : siehe 'AReq.batch'

(Thomas Knoth/jb)

Große Schrift mit Notepad

Ist es Ihnen nicht manchmal auch so ergangen, daß Sie in irgendeiner Form mal eine nettere und größere Schrift benötigten, ohne vorher umständliche Schriftenprogramme zu laden oder sich durch Malprogramme zu quälen?
Das Notepad macht es, besonders da jetzt ja der "Ausbügeleffekt" besteht. Schön und gut. Ich jedenfalls benötigte nur mal so ganz einfach eine größere Menge von groß beschrifteten Aktenordner-Rükken. Der Effekt unter dem 'Projekt = Print Large' war nicht so übel, doch noch immer zu klein. Selbst mit dem größtmöglichen Font (Diamont 20) hat es mich nicht vom Hocker gerissen. Und dann kam mir eine Idee. Ich verkleinerte das Notepad-Fenster, so daß gerade noch die gewünschte Schrift zum Vorschein kam und dabei passierte es, daß die Schrift so vergrößert wurde, daß ich zufrieden war. Wem kann dies helfen? Ich spreche hier besonders die Schüler an, die sich nicht gleich ein teueres Malprogramm kaufen können, hier haben Sie mit einfacher und simpler Methode einen kleinen Zeichenclou.
P.S.: Wer jetzt übertreiben will, legt noch eine Font-Diskette ein, die eine extra große Schrift (Helvetica 40) bietet. Doch dann können Sie gleich Verhandlungen mit einer Firma führen, die Litfaßsäulen-Papier liefert!

(Gerd Hunger/jb)

Besseres INPUT in GFA-Basic

Der am meisten gebrauchte Befehl, um Eingaben vom Benutzer zu holen, ist sowohl in Amiga-BASIC als auch in anderen BASIC-Dialekten der mehr oder weniger komfortable Befehl INPUT. Der INPUT-Befehl in GFA-BASIC hat seinem Amiga-BASIC-Kollegen einiges voraus:
a) In GFA-BASIC kann man den zur Eingabe auffordernden Text nicht mehr einfach mit der Backspace-Taste löschen.
b) In GFA-BASIC kann man mit den Cursor-Tasten im Text herumspringen und so sehr viel einfacher Korrekturen vornehmen.
Allerdings hat auch der GFA-BASIC-INPUT-Befehl zwei Schwächen:
a) Nach dem zur Eingabe auffordernden Text erscheint immer das lästige Fragezeichen, zum Beispiel: INPUT "Eingabe:";ein$ ergibt auf dem Bildschirm 'Eingabe:?'.

b) Wenn man in einer durch INPUT erzeugten Abfrage Text eingibt und die BACKSPACE-Taste betätigt, wird rechts von der Eingabe befindliche Grafik mit dem Cursor nach links gezogen:

Grafiklinie
▮
Eingabe:? Testeingb ▮
▮

Der Benutzer hat sich vertippt und betätigt die BACKSPACE-Taste:

▮
Eingabe:? Testeing ▮
▮

Selbst wenn jetzt der berichtigte Text weiter eingegeben wird, wird die Grafik nicht wieder nach rechts eingerückt.

ABHILFE:
a) Um das lästige Fragezeichen loszuwerden, muß nur die Syntax des INPUT-Befehls etwas geändert werden:

INPUT "Eingabe:",ein$

Wird also statt des Semikolons ein Komma benutzt, wird das Fragezeichen unterbunden.
b) Die Lösung des zweiten Problems ist etwas komplizierter. Hier kann man leicht auf ein Hilfsprogramm zurückgreifen , wenn der Problemfall vorliegt. Wie gesagt, es ist nur ein Hilfsprogramm, bei dem, um die Übersichtlichkeit so groß wie möglich zu halten, auf die Bedienung durch die Cursor-Tasten verzichtet werden mußte. Dafür verfügt SMART_INPUT, so heißt unser Hilfsprogramm, über andere Optionen, die die

Wichtiger Hinweis:

Da manche Listings, Dateien und Listen zu sehr in die Breite gehen, sind wir manchmal gezwungen, Zeileneinrückungen fallenzulassen, um mit der Breite einer Textspalte nicht in Konflikt zu kommen. Damit Sie beim Eingeben keinerlei Probleme haben, werden alle Dateien mit Zeilennummern versehen. Diese dienen nur der besseren Übersicht und sollten nicht mit übernommen werden. Bei der Eingabe der Dateien halten Sie sich bitte an die entsprechenden Vorgaben, für die jeweiligen Programme. Mountlists sollten so eingegeben werden, wie sie im DEVS-Verzeichnis stehen. (Red.)

Eingabe eines Textes komfortabler gestalten:
Das mit MERGE einzubindende Unterprogramm 'SMART_INPUT' verlangt sechs Parameter. Der Aufruf sieht so aus:

SMART_INPUT(x-Position%,y-Position%,Text$,MIN%,MAX%,geheim%)

Zu den Parametern:

x-Position: die x-Koordinate, an der der Text erscheint, der zur Eingabe auffordert (Integer)
y-Position: die y-Koordinate, eine Einheit wie Spalte, Zeile (Integer)
Text: Text, der zur Eingabe auffordert, zum Beispiel "Ihr Name:" (String)
MIN: vom Programmierer festgelegte Mindestlänge des einzugebenden Textes (Integer)
MAX: vom Programmierer festgelegte Höchstlänge des einzugebenden Textes (Integer)
geheim: eine Option, die es erlaubt, Text verdeckt einzugeben – bei der Eingabe erscheint statt des getippten Zeichens nur ein X (beliebig änderbar); um Text verdeckt einzugeben, muß die Variable 'geheim%' im Parameter auf TRUE (-1) gesetzt werden (Integer).
Das Unterprogramm SMART_INPUT kann den individuellen Bedürfnissen des Benutzers angepaßt werden, indem man zum Beispiel das Cursor-Zeichen oder die ausgegebenen Zeichen im Geheim-Modus ändert. Ein weiterer GFA-BASIC-Befehl zur Texteingabe ist der Befehl INPUT$ mit der Syntax:

ein$ = INPUT$(anzahl)

Dieser Befehl liest 'anzahl' Zeichen von der Tastatur und legt sie im String 'ein$' ab. Während der gesamten Eingabe wird kein Zeichen auf dem Bildschirm ausgegeben. Dieses Beispiel der verdeckten Texteingabe hat aber auch einen Nachteil, der durch SMART_INPUT gelöst werden kann: Wenn die durch 'anzahl' bestimmte Höchstmenge einzugebender Zeichen erreicht ist, behandelt GFA-BASIC die Eingabe als abgeschlossen und beendet die Eingabe, ohne dem Benutzer die Möglichkeit zu geben, seinen Text zu verbessern (blind). SMART_INPUT schafft auch hier Abhilfe: Wenn man im Programm das Zeichen 'X' durch ein Leerzeichen <SPACE> ersetzt und im Parameter den Geheim-Modus aktiviert (durch TRUE), so wird der String ebenfalls verdeckt eingegeben, ohne daß man sehen könnte, wieviele Zeichen getippt wurden. Allerdings bleibt dem Benutzer auch nach Erreichen der maximalen Zeichenanzahl immer noch die Möglichkeit, seine Eingabe zu verbessern. Außerdem gibt es vor allem einem Anfänger das Gefühl, das Programm besser unter Kontrolle zu haben, als wenn der Computer die Eingabe selbständig beendet.

(Boris Funke/jb)

Schneller mit der RAM-Disk

Wenn man mit dem CLI arbeitet, muß man immer etwas länger warten, da die entsprechenden Befehle erst von der Workbench geladen werden müssen. Wenn man nun aber die Befehle in die RAM-Disk kopiert, kann man mit dem CLI um einiges schneller arbeiten. Man kopiert das Verzeichnis C der Workbench so in die RAM-Disk (für Amiga mit 512 kByte): *makedir ram:c*

(Verzeichnis C im RAM erstellen)
copy c/copy to ram:c
copy c/dir to ram:c
copy c/cd to ram:c
copy c/info to ram:c
copy c/list to ram:c

...und noch weitere Befehle, die man hin und wieder braucht. Hat man 1 MByte Speicher zur Verfügung, kann man das gesamte C-Verzeichnis ins RAM kopieren:

copy c to ram:c

(Sascha Endlein/jb)

Guter Rat fürs RAD:!

Für die optimale Nutzung ihrer RAD: sollten sie das FAST-FILING-SYSTEM (Beschleunigung des Zugriffs) nutzen. Hierzu sind lediglich einige Änderungen in der Mountlist (Verzeichnis "devs") notwendig. Achten Sie bitte darauf, das Ihre Bootpriorität auf -128 festgelegt ist und Surfaces, BlockPerTrack, LowCyl und HighCyl den Werten eines Diskettenlaufwerks entsprechen. Die RAD: in der Mountlist sieht nun wie folgt aus:

```
 1 RAD: Device = ramdrive
                 .device
 2 Unit = 1
 3 Flags = 0
 4 Surfaces = 2
 5 BlockPerTrack = 11
 6 Reserved = 2
 7 Interleave = 0
 8 LowCyl = 0 ; HighCyl = 79
 9 Buffers = 5
10 BufMemType = 1
11 GlobVec = -1
12 StackSize = 4000
13 DosType = 0x444F5301
14 FileSystem = L:fastfilesy
                 stem
15 BootPri = -128
16 #
```

(Die Zahlen vor den Zeilen dürfen natürlich nicht mit übernommen werden, sie dienen nur der Übersichtlichkeit, Red.)

Aufgrund der Bootpriorität von -128 ist nun kein Autoboot mehr möglich.

REM-Ersatz

In Amiga-BASIC wird wie in anderen BASIC-Dialekten auch für Anmerkungen der Befehl REM (REMark) verwendet. Man kann den gleichen Befehl aber auch durch Setzen eines Apostrophs erreichen. Der Editor erkennt allerdings nur das Apostroph, das über die Tastenkombination <ALT> + <ä> aufgerufen wird, als ein REM an; die auf der Tastatur vermerkten funktionieren hierbei nicht.

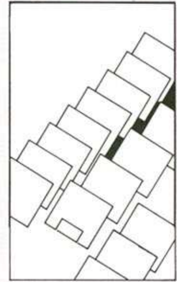
(Christian Czerwinski)

Icons für PD-Disketten

Bei älteren Public-Domain-Disketten kann ab und zu das Problem auftauchen, daß die enthaltenen Programme sich nicht von der Workbench laden lassen, da die entsprechenden Programme keine Icons besitzen und man sie somit nicht komfortabel starten kann. Mein kleines Programm sorgt hier für Abhilfe, obwohl es nur sieben Zeilen lang ist. Geschrieben wurde es in GFA-BASIC und sollte möglichst mit dem GFA-BASIC-Compiler compiliert werden, damit es sinnvoll eingesetzt werden kann. Danach ist es in die Startup-Sequence einzubinden, damit später die entsprechenden Programme mit der Maus gestartet werden können. Hier nun das Programm:

```
1 start:
2 fileselect "SYS:","EXE-
  CUTE"
  ,"",la$
3 if la$ = "" then
4 end
5 endif
6 exec la$,-1,-1
7 goto start
```

(Manfred Dittrich jun./jb)



Tausche Tip gegen DATABOX

Haben Ihnen die Kurz-Tips auf unseren Seiten gefallen? Vielleicht haben Sie selbst einen Tip parat, der andere Amiga-Besitzer interessieren könnte? Dann machen wir Ihnen wieder unser Angebot: Sie schicken einen oder mehrere kurze Tips an uns, und wenn diese(r) uns gefällt(allen) , bekommen Sie von uns im Austausch die DATABOX desjenigen Heftes zugeschickt, in dem Ihr Tip erscheint.
Dabei sollten Sie noch folgendes beachten:

– Schreiben Sie Ihren Tip als ASCII-Datei auf Ihre Diskette, möglichst ohne eigensinnige Verschnörkelungen, da alle Texte zur weiteren Verarbeitung auf Personal-Computer transferiert werden müssen. Sonder- und Steuerzeichen haben dabei meist seltsame Auswirkungen.

– Schreiben Sie auf den Brief sowie auf einen kleinen Aufkleber auf der Diskette Ihre lückenlose Anschrift und einen kleinen Hinweis, was auf der Diskette zu finden ist.

Dann können Sie sicher sein, daß die DATABOX Sie erreicht. Schicken Sie das Ganze an den

DMV-Verlag
Redaktion AMIGA DOS
Kennwort "Tausche Tip gegen Databox"
Postfach 250
D-3440 Eschwege

Listings

```
PROCEDURE smart_input(x%,y%,text$,min%,max%,geh%)
' x-Koordinate, y-Koordinate, Requester-Text, minimale Eingabelänge
' maximale Eingabelänge, GEHEIM-Modus
 LOCAL a%,k%                    ! lokale Variablen
 PRINT AT(x%,y%);text$;" _";    ! Cursor auf Anfang setzen
 x%=x%+LEN(text$)+1             ! Eingabestring initialisieren
 ein$=SPACE$(max%)              ! Tastaturpuffer leeren
 WHILE INKEY$<>""
WEND                            ! S C H L E I F E
 DO                             ! Zeichen von Tastatur holen
   i$=INKEY$                    ! ASCII-Wert bilden
   a%=ASC(i$)                   ! Backspace eingegeben UND Eingabestring
   IF a%=8 AND k%>0             !  mindestens ein Zeichen lang
     DEC x%
     MID$(ein$,k%,1)=" "        ! Cursor nach links versetzen und String-
     DEC k%                     !  zeiger verschieben
     PRINT AT(x%,y%);"_ ";
     GOTO weiter2
   ELSE IF a%=196 OR a%=214 OR a%=220 OR a%=228 OR a%=246 OR a%=252
     GOTO weiter1               ! weiter, wenn Umlaut eingegeben
   ELSE IF a%<32 OR a%>125      ! sonst - kein gültiges ASCII-Zeichen
     GOTO weiter2
   ENDIF
   weiter1:                     ! wenn Anzahl der erlaubten Zeichen noch
   IF k%<max%                   !  nicht erreicht, dann weiter
     IF geh%=TRUE               ! wenn verdeckt eingegeben werden soll,
       PRINT AT(x%,y%);"X";     !  nur 'X' ausgeben
     ELSE
       PRINT AT(x%,y%);i$;      ! sonst eingegebenes Zeichen zeigen
     ENDIF
     IF k%+1<max%               ! wenn noch nicht vorletzte Stelle, Cursor
       PRINT "_"                !  nach rechts versetzen
     ENDIF
     INC x%                     ! x-Koordinate des Cursors und
     INC k%                     ! Anzahl der Zeichen um 1 erhöhen
     MID$(ein$,k%,1)=i$         ! Eingabestring um eingegebenes Zeichen
     weiter2:                   !  erweitern
   ENDIF
   EXIT IF i$=CHR$(13) AND k%=>min% ! Schleife verlassen, wenn RETURN ge-
 LOOP                           ! drückt wurde UND mindestens min% Zeichen
 '                              !  eingegeben wurden
 PRINT AT(x%,y%);" "            ! letzten Cursor löschen (wichtig, wenn
                                ! weniger als max% Zeichen eingegeben
                                ! wurden, da sonst ein Cursor-Zeichen
                                ! stehen bliebe
 ein$=TRIM$(ein$)               ! eventuelle Leerzeichen vom Eingabestring
RETURN                          !  trennen
```

Listing:SMART_INPUT von Boris Funke
Die Zeilennummern dienen nur der besseren Uebersicht!

```
/*        Programm: AReq                                  *
 * Compiler: Lattice Amiga C Compiler V5.02               *
 * Autor   : Thomas Knoth                                 *
 *                                                        *
 * Copyright ~ 1989, 1990 by Th. Knoth, Bremen            *
 *                                                        */

#include <proto/dos.h>
#include <dos.h>
#include <proto/exec.h>
#include <intuition/intuition.h>

struct IntuitionBase *IntuitionBase;

BOOL   req(char *,char *, char *);

struct IntuiText Kasten[3] =
{
  0, 1, JAM1,10,10, NULL, NULL, NULL,
  2, 3, JAM1, 5, 3, NULL, NULL, NULL,
  2, 3, JAM1, 5, 3, NULL, NULL, NULL
};

void main(argc, argv)
SHORT argc;
signed char *argv[];
{
  if(argc==4)
  {
    IntuitionBase=(struct IntuitionBase*)OpenLibrary("intuition.library", 0L);

    if((req(argv[1],argv[2], argv[3]))==TRUE)
    {
      CloseLibrary(IntuitionBase);
      exit(RETURN_WARN);
    }
    else
    {
      CloseLibrary(IntuitionBase);
      exit(RETURN_OK);
    }
  }
  else
  {
    (void)Write(Output(),"Aufruf : AReq Body LGadget RGadget\n",35);
    (void)Write(Output(),"Copyright (c) 1989, 1990 by Th. Knoth, Bremen\n",45);
    exit(RETURN_ERROR);
  }
}

BOOL req(title, yes, no)
char    *title,*yes,*no;
{
  Kasten[0].IText=(UBYTE *)title;
  Kasten[2].IText=(UBYTE *)no;
  Kasten[1].IText=(UBYTE *)yes;
  return((BOOL)AutoRequest(NULL, &Kasten[0], &Kasten[1], &Kasten[2], 0L, 0L, 320L, 60L));
}
```

Listing:AReq.C von Thomas Knoth

```
; Beispiel Batch-Datei zum Befehl "areq"
; Startet areq und fragt danach den Fehlercode ab.
; areq gibt warn zurueck, wenn das linke Gadget gedrueckt wurde, in diesem
; Fall "Natuerlich". Ansonsten wird der Fehlercode "0" (alles ok) zurueck-
; gegeben.
; areq kehrt mit dem Fehlercode 10 zurueck, wenn ein falscher Aufruf er-
; folgte !
;
; Copyright (c) 1989, 1990 by Th. Knoth, Bremen

areq "Benutzen Sie die WB 1.3 ?" Natuerlich "NOCH nicht !"
if warn
  echo "Er ist UptoDate !"
else
  echo "Last but not least ein Oldie !"
endif
```

Listing:AReq- Test-Batch-File

```
.key datei

lc1 <datei$tl>.c
go <datei$tl>.q
lc2 <datei$tl>.q
blink lib:cres.o <datei$tl>.o to <datei$tl>.res lib lib:lc.lib,lib:amiga.lib

; Copyright (c) 1989, 1990 by Th. Knoth, Bremen
```

Listing:MAKE.RES-Batch-Datei

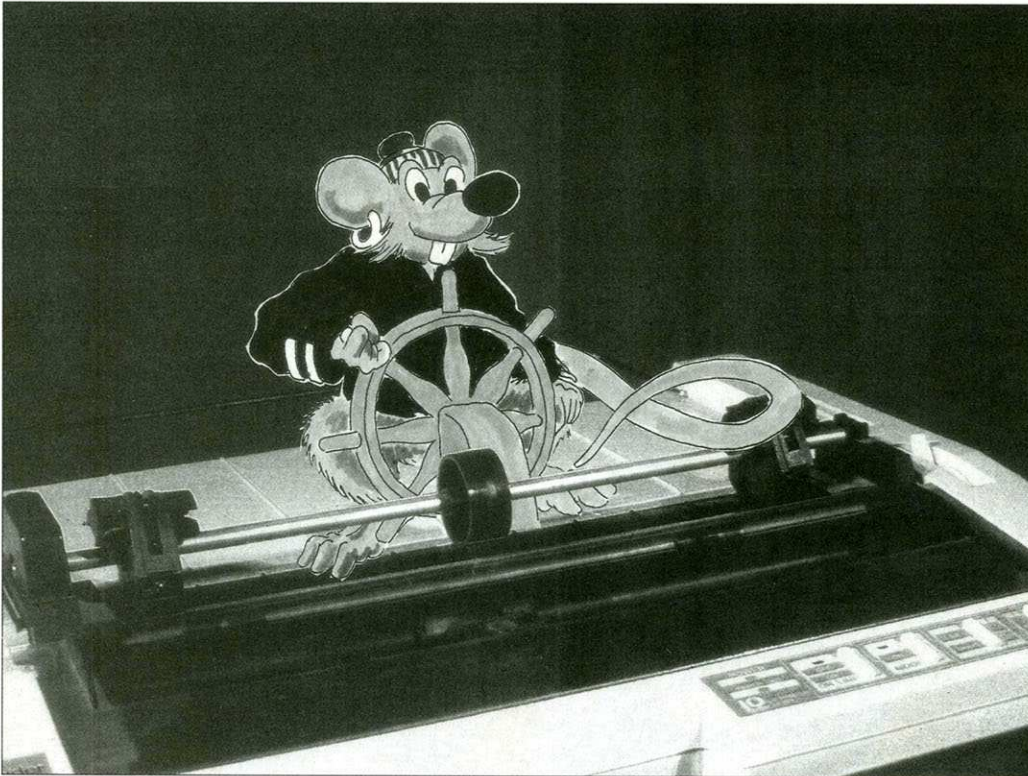
```
.key datei

lc1 <datei$tl>.c
go <datei$tl>.q
lc2 <datei$tl>.q
blink lib:c.o <datei$tl>.o lib lib:lc.lib,lib:amiga.lib

; Copyright (c) 1989, 1990 by Th. Knoth, Bremen
```

Listing:AReq-Make-Batch-File
Bitte die Batch- Dateien ohne Zeilennummern mit Editor eingeben!

Listing: die 'Gewußt wie!'-Programme



Garry Glendown

Ruft die Mäuse auf den Plan

Printer Control Window – Druckersteuerung per Maus

Sie kennen folgendes Problem: Oftmals hat man eben mal zwei oder drei Seiten auszudrucken, für die man ein paar Control-Codes benötigt, oder man möchte eine besondere Randeinstellung für das Ausdrucken eines Listings haben. Hat dieses File dann noch einige Standard-Epson-Controlcodes, die andere Druckereigenschaften beeinflussen, fällt die Benutzung von PRT flach. Was also tun, sprach Zeus...? Ganz einfach: nur PCW mit Run starten, die benötigten Optionen anklicken, und schon ist der Drucker einsatzbereit.

Kommen wir auf das Menü etwas näher zu sprechen. Ein Eintrag in der Menüliste besteht aus mehreren Teilen. "Type" ist die Art des Eintrags. Im Moment existieren folgende Typen:

- pr_COMMAND

Die in dem Array 'Prt' abgelegten Daten sind auszudruckende Daten, die Länge steht in 'PLen'.

- pr_EXTERN

Es wird ein externer Befehl ausgeführt, dessen Nummer in 'PLen' steht.

- pr_NONE

Auf Anwahl dieses Menüpunkts erfolgt keine Reaktion, er dient lediglich der Information.

"Name" ist ein String, in dem der zu druckende Text steht. Der Text darf maximal 'MAXLINE'-Zeichen enthalten.

"Len" ist die Länge des zu druckenden Strings und wird vom Programm eingefügt (in der Routine 'OpenStuff()', siehe Remarks).

"Prt" ist ein Array, das die zu druckenden Zeichen enthält.

"PLen" gibt die Anzahl der zu druckenden Zeichen bzw. die Nummer der Extern-Routine wieder.

"FPen" und "BPen" stellen die Vorder- und Hintergrundfarbe des Textes im Menü dar.

Die Anzahl der Menüpunkte muß ein Vielfaches von sechs sein, da jeweils sechs Menüpunkte auf einmal angezeigt werden. Man kann zwar das Menüprogramm so umändern, daß es beim Scrollen um jeweils eine Zeile nach oben bzw. unten geschoben wird, aber dies schien mir, vor allem in Anbetracht der teilweise doch recht großen Datenmenge, uninteressant, da das Blättern nach einem bestimmten Menüpunkt zu lang dauert.

Nicht nur Maus – nein, auch Taste

Alle Menüpunkte können auch über die Tastatur angewählt werden. Dabei gilt folgende Belegung:

- Tasten 1 bis 6: Auswahl der einzelnen Menüpunkte
- Crsr Dn: Blättern nach unten
- Crsr Up: Blättern nach oben
- Help: Info
- Q: Programm beenden (Requester beachten!)

Sollten bei der Initialisierung Fehler auftreten, so wird der Benutzer mittels eines Requesters darauf hingewiesen. Außer dem Requester, der bei Anwählen von 'Q' bzw. der Exit-Funktion erscheint, ist es egal, mit welchem Gadget man den Requester quittiert. Sollte man die Exit-Funktion (bzw. 'Q' auf der Tastatur) aus Versehen angewählt haben, so kann man im Programm bleiben, indem man 'Nein' anwählt (Tastatur: LeftAmiga-V), durch Selektieren von 'OK' (LeftAmiga-B) wird PCW verlassen.

Auf, ans Eingemachte

Deaktiviert man das PCW-Window (durch Anklicken eines anderen Fensters), wird es in Y-Richtung auf wenige Pixel verkleinert, um so wenig Platz wie möglich auf dem Bildschirm zu belegen. Man kann es natürlich auch mittels Depth("Tiefen")-Gadget hinter andere Windows legen, doch ist es recht praktisch, wenn es jederzeit ohne große Sucherei anklickbar ist. Durch die Verkleinerung nimmt es in Y-Richtung genausoviel Platz ein wie die Menüzeile eines Windows, so daß man es bequem in die linke oder rechte obere Ecke legen kann.

Im Zusammenhang mit dem Vergrößern des Windows auf Normalgröße fällt eine Eigenheit von Intuition auf, die ich schon als 'Bug' bezeichne: Es kommt nämlich die IntuiMessage 'ACTIVEWINDOW', wenn das Window verschoben wird und vorher inaktiv war. Diese Meldung kommt aber leider zu früh, sofort nach Anklicken des Windows, obwohl das Window noch bewegt wird. Greift man dann auf Teile der Window-Struktur zu, so enthält sie die (für diese Verwendung) falschen Werte.

Das führt dann dazu, daß das Window aus dem Bildschirm heraus'gesized' wird, was meist zu einem nicht so schö-

nen Bildaufbau führt (oder wie es im Addison-Wesley-Buch heißt: "Wegen Verzerrungen im Raum-Zeit-Kontinuum, die daraus resultieren können, wie es in der speziellen Relativitätstheorie beschrieben ist, bietet das Ergebnis gewöhnlich keinen schönen Anblick"). Daher ist im Programm darauf zu warten, daß der linke Mausknopf losgelassen wird, um erst danach (wenn das Window an seiner endgültigen Position liegt) auf die Window-Koordinaten zuzugreifen. Zusätzlich dazu wartet man im Verlauf der Size-Routine noch mehrmals auf Intui-Ticks, also jeweils n/10 Sekunden, um sicherzugehen, daß die Operationen wie MoveWindow und SizeWindow fertig durchgeführt sind.

Jetzt passiert's ...

Wird ein Menüpunkt ausgewählt, ruft das Hauptprogramm die Routine 'DoIt' zusammen mit der Nummer der Auswahl auf. Diese Routine wickelt dann alle anderen Funktionen ab. Somit beschränkt sich das Hauptprogramm lediglich auf das Abholen und Auswerten der IntuiMessages.

"DoIt" verzweigt als erstes nach dem Typ der Auswahl, also "pr_Command", "pr_Extern" oder "pr_None" (letzteres wird eigentlich einfach übergangen). Handelt es sich bei der Auswahl um einen Druckerbefehl, so wird der Druckerkanal geöffnet, der String ausgegeben, ein "BEEP" (CHR$(7)) als Bestätigung des Kommandos angehängt und der Druckerkanal wieder geschlossen. Im Falle von "pr_Extern" wird nach der Nummer von "men[n].PLen" verzweigt. In dem abgedruckten Programm steht dabei '1' für Quit, '2' für Info und '3' für Rückkehr ins normale Menü.

Hier kann das Programm natürlich beliebig erweitert werden, zum Beispiel durch Einfügung des Ausdrucks Files oder anderen Utilities. Die einzige Beschränkung besteht darin, daß man nur 256 verschiedene Extern-Routinen definieren kann, da "PLen" als BYTE definiert ist (Klar, man kann das in 'Int' oder gar 'long' ändern, aber ich habe noch kein Programm mit über vier Milliarden Optionen gesehen; abgesehen davon bedeutet das Optionen von 6 Seiten. Viel Vergnügen beim Durchblättern...). Auch ohne Änderung von "PLen" kann man noch mehr Routinen einbauen, nämlich durch Einführen eines neuen Typs, zum Beispiel "pr_Extern2" (Wer mehr als 512 Funktionen benötigt, sollte mal seinen Programmierstil überprüfen oder sich eine(n[?]) Cray 2 anschaffen...).

Hosen runter...

Wie im Skat muß auch hier das Blatt aufgedeckt werden, allerdings handelt es sich hier um die Menüpunkte. Dies geschieht in der "Display_em"-Routine. Sie übernimmt alles von der Positionierung des Cursors über die Farbeinstellung bis hin zum Ausdrucken der Texte. Vorher wird der Bildschirmbereich gelöscht.

Zum Schluß noch ein Hinweis: Alle Drucker-Codes im abgedruckten Programm gelten für Drucker der NEC-Pinwriter-Serie sowie Kompatible (Epson Q-Serie etc.). Das heißt, falls Sie Ihren Drucker mit einem Treiber wie "CBM_MPS2xxx", "Epson_Q" oder "NEC_Pinwriter" betreiben, können Sie das Programm ohne Änderung übernehmen.

Andernfalls sollten Sie versuchen, die Codes entweder direkt mit dem Programm auf korrekte Funktion zu testen, oder aber die richtigen Codes mittels Blick ins Handbuch einsetzen.

(vb)

Listings

```
/**********************************************************/
/*                      prcw.c                            */
/*            Printer Control Window V 2.0                */
/*        (C) 1990 by G.Glendown & AMIGA DOS              */
/*                 Sprache: Aztec C                       */
/**********************************************************/

/*
Achtung! Die  in  diesem Programm verwendeten Drucker-
Codes  sind gültig  für  die NEC-Pinwriter-Drucker und
Kompatible. Bei anderen Druckern können einzelne Codes
falsch oder gar nicht funktionieren. Konsultieren Sie
daher  im Zweifelsfall ihr Handbuch, um zu sehen, ob
und wie Ihr Drucker die abgedruckten Funktionen ver-
wendet.
Compilier-Hinweis:  Alle  Includes sind vorcompiliert,
das heißt, in einem File abgelegt, so daß lediglich
ein File eingelesen werden muß. Um dieses File zu er-
zeugen, muß man ein Source-File  schreiben, das alle
Include-Files mit #include einliest und dieses mit:
          cc <name>.c -a +h<include-name>
compilieren. Das fertige File wird dann beim Compi-
lieren mit:
          cc <prg>.c +i<include-name>
eingebunden. Zum Linken wird lediglich die c.lib-
library benötigt.                                        */


#define UBP           (UBYTE *)
#define pr_COMMAND    0x01
#define pr_EXTERN     0x02
#define pr_NONE       0x00
#define ESC           27
#define FS            28
#define MAXLINE       21

#include "pcw_defs.h"

struct prt {
     BYTE     Type;              /* Art des Eintrags:
                                    pr_Command/Extern/None */
     UBYTE    Name[MAXLINE]; /* Auszudruckender Name  */
     BYTE     Len;               /* Länge des Texts      */
     UBYTE    Prt[10];           /* Prt-Codes            */
     BYTE     PLen;              /* Länge Prt-Codes
                                    bzw. Extern-Routine  */
     UBYTE    FPen,BPen;
                      /* Vorder-/Hintergrundfarbe        */
     };

struct prt info[] = {
                   /* Info, wird als Menü gehandhabt.
                      Sobald ein Menüpunkt ausgewählt
                      wird, wird in den normalen Teil
                      zurückgeschaltet.                  */

                   /* Extern Routine #3:
                      Rückkehr ins normale Menü          */
     pr_EXTERN,"Dieses Programm ist",0,"",3,1,0,
     pr_EXTERN," komplett mausge-",0,"",3,1,0,
     pr_EXTERN," steuert. Zum Aus-",0,"",3,1,0,
     pr_EXTERN," wählen der Seite",0,"",3,1,0,
     pr_EXTERN," einfach einen der",0,"",3,1,0,
     pr_EXTERN," Pfeile anklicken!",0,"",3,1,0,

     pr_EXTERN,"Zum Auswählen einer",0,"",3,1,0,
```

Listing: Printer Control Window V2.0

```
     pr_EXTERN," Option diese nur",0,"",3,1,0,
     pr_EXTERN," mit der Maus an-.",0,"",3,1,0,
     pr_EXTERN," klicken. Rückkehr",0,"",3,1,0,
     pr_EXTERN,"zum Hauptmenü durch",0,"",3,1,0,
     pr_EXTERN," irgendeine Option!",0,"",3,1,0,

     pr_EXTERN,"",0,"",3,1,0,
     pr_EXTERN,"    Alle Optionen",0,"",3,1,0,
     pr_EXTERN,"können auch mit der",0,"",3,1,0,
     pr_EXTERN,"  Tastatur gewählt",0,"",3,1,0,
     pr_EXTERN,"        werden:",0,"",3,1,0,
     pr_EXTERN,"",0,"",3,1,0,

     pr_EXTERN,"Taste    Funktion:",0,"",3,1,0,
     pr_EXTERN,"1-6      Menüpunkt",0,"",3,1,0,
     pr_EXTERN,"Crsr up Seite -",0,"",3,1,0,
     pr_EXTERN,"Crsr dn Seite +",0,"",3,1,0,
     pr_EXTERN,"Q        Ende",0,"",3,1,0,
     pr_EXTERN,"",0,"",3,1,0
};
#define NUM_INFO     4

struct prt menu[] = {
     pr_COMMAND,"Italic on",0,{ESC,'4'},2,1,0,
     pr_COMMAND,"Italic off",0,{ESC,'5'},2,1,0,
     pr_COMMAND,"Double strike on",0,{ESC,'G'},2,1,0,
     pr_COMMAND,"Double strike off",0,{ESC,'H'},2,1,0,
     pr_COMMAND,"Underline On",0,{ESC,'-',1},3,1,0,
     pr_COMMAND,"Underline Off",0,{ESC,'-',0},3,1,0,

     pr_COMMAND,"10 CPI",0,{18,ESC,'P'},3,1,0,
     pr_COMMAND,"12 CPI",0,{18,FS,'S',0,ESC,'M'},6,1,0,
     pr_COMMAND,"High Speed",0,{18,ESC,'M',FS,'S',1},6,
 1,0,
     pr_COMMAND,"15 CPI",0,{18,ESC,'g'},3,1,0,
     pr_COMMAND,"17 CPI",0,{ESC,'P',15},3,1,0,
     pr_COMMAND,"20 CPI",0,{ESC,'M',15},3,1,0,

     pr_COMMAND,"CS: USA",0,{ESC,'R',0},3,1,0,
     pr_COMMAND,"CS:  D",0,{ESC,'R',2},3,1,0,
     pr_COMMAND,"CS: IBM",0,{FS,'I',1},3,1,0,
     pr_COMMAND,"6 LPI",0,{ESC,'2'},2,1,0,
     pr_COMMAND,"8 LPI",0,{ESC,'0'},2,1,0,
     pr_COMMAND,"LF: 1/18",0,{ESC,'3',10},3,1,0,

     pr_COMMAND,"Superscript",0,{ESC,'S',0},3,1,0,
     pr_COMMAND,"Subscript",0,{ESC,'S',1},3,1,0,
     pr_COMMAND,"Cancel Sub/Superscr.",0,{ESC,'T'},2,1,
 0,
     pr_COMMAND,"FF",0,{12},1,1,0,
     pr_COMMAND,"SO on",0,{ESC,'W',1},3,1,0,
     pr_COMMAND,"SOáoff",0,{ESC,'W',0},3,1,0,

     pr_COMMAND,"Left Justify",0,{ESC,'a',0},3,1,0,
     pr_COMMAND,"Right Justify",0,{ESC,'a',1},3,1,0,
     pr_COMMAND,"Center Justify",0,{ESC,'a',2},3,1,0,
     pr_COMMAND,"Full Justify",0,{ESC,'a',3},3,1,0,
     pr_COMMAND,"Proportional on",0,{ESC,'p',1},3,1,0,
     pr_COMMAND,"Proportional off",0,{ESC,'p',0},3,1,0,

     pr_COMMAND,"Draft",0,{ESC,'x',0},3,1,0,
     pr_COMMAND,"LQ",0,{ESC,'x',1},3,1,0,
     pr_COMMAND,"Half Speed",0,{ESC,'s',1},3,1,0,
     pr_COMMAND,"Full Speed",0,{ESC,'s',0},3,1,0,
     pr_COMMAND,"User-defined CS on",0,{ESC,'%',1},3,1,
 0,

     pr_COMMAND,"User-defined CS off",0,{ESC,'%',0},3,1
 ,0,
```

Listing: Printer Control Window V2.0

```
     pr_NONE,   "PrinterControlWindow",0,{},0,0,1,
     pr_NONE,   "--------------------",0,{},0,0,1,
     pr_NONE,   "1990 by G.Glendown",0,{},0,0,1,
     pr_COMMAND,"Reset Printer",0,{FS,'@'},2,1,0,
     pr_EXTERN,"      ---HELP---      ",0,{},2,3,1,
                                   /* Extern-Routine #2 */
     pr_EXTERN,"       Q U I T       ",0,{},1,2,3
                                   /* Extern-Routine #1 */
     };

#define NUM_MENS     7

struct GfxBase           *GfxBase;
struct IntuitionBase     *IntuitionBase;
struct RastPort          *rp;
struct FileHandle        *fh,*Open();
struct IntuiMessage      *msg,*GetMsg();
struct Window            *win,*OpenWindow();

struct Gadget            *gadgadr;
int     gid;
long    class;
int     code;      /* IntuiMsg-Decode                 */
int     max;       /* Maximale Anzahl Menü-Seiten     */
struct  prt      *men;
                   /* Zeiger auf aktuelle Menü-Liste */

char    Beep[2]={7,7};                 /* Für Drucker */

int     blk=-1,nb=0,slct;
                   /* Aktueller 6er-Block, neuer 6er-
                      Block selektierte Funktion     */

main()
{
int temp,slt;
long cl;
int keypressed=0;
     OpenStuff();              /* Libs & Window öffnen  */
     men=menu;                 /* Menü : Hauptmenü      */
     max=NUM_MENS;
     for (;;) {
          if (nb!=blk) {

                    /* Seitennummer hat sich geändert,
                       Menü muß neu angezeigt werden. */
               blk=nb;
               Display_em();
          }
          WaitPort(win->UserPort);

                   /* Damit keine Busy-Wait Schleife:
                     Task ist im Status 'Wait' und
                     verbraucht keine Prozessor-Zeit */

          msg=GetMsg(win->UserPort);
          class=msg->Class;
          code=msg->Code;
          gadgadr=(struct Gadget *)msg->IAddress;
          ReplyMsg(msg);

          switch (class) {
               case RAWKEY:

     if (code>127) keypressed=0;
                    if (!keypressed) {
                  if ((code>0)&&(code<7))
        DoIt(blk*6+code-1),keypressed=1;
                  else {
                      switch (code) {
                          case 76:      /*   Cursor up */
                          case 77:      /*   Cursor dn */
                           keypressed=1;
                              code*=2;
                              nb=blk+code-153;
                              if (nb>=max) nb=0;
                              if (nb<0) nb=max-1;
                              break;
                             case 95:   /*    Help     */
                              men=info;
                              keypressed=1;
                              max=NUM_INFO;
                              blk=-1;
                              nb=0;
                              break;
                          case 16:
                           keypressed=1;
                           ex(1);
                           break;
                      }
                  }
                 }
            break;
               case INACTIVEWINDOW:
                    if (win->Height>86)
                        SizeWindow(win,0L,-76L);
                    break;
               case ACTIVEWINDOW:
                    if (win->Height<88) {

                       /* Warten, bis linke Maustaste los-
                          gelassen wird.                 */
                       /* Anmerkung: Könnte weggelassen
                          werden, wenn Intuition die
                          ACTIVEW.-Message erst nach dem
                          Loslassen der Drag-Bar schicken
                          würde ...                      */
#asm
                     move.l  d0,-(a7)
lbl:                 move.b  $bfe001,d0
                     and.b   #$40,d0
                     beq.s   lbl
                     move.l  (a7)+,d0
#endasm
                     WaitIT(3);
                     temp=
                     win->WScreen->Height-88-win->TopEdge;
                        if (temp<0) {
                       /* Window ist zu weit unten um es
                          zu vergrößern, also noch oben
                          schieben...                    */

                          WaitIT(5);
                          MoveWindow(win,0L,(long)temp);
                          WaitIT(5);
                        }
                       SizeWindow(win,0L,76L);
                       WaitIT(5);
                       RefreshGadgets(&Gadget1,win,NL);
                       Display_em();

                    }
                    break;
               case GADGETUP:
                    gid=gadgadr->GadgetID;

                    if (gid<30) {
                       /* Einer der normalen Menüpunkte */
                        slct=blk*6+gid-21;
                        DoIt(slct);
                    }

                    else {
                       /* einer der Pfeile oder 'Info'  */
                        gid-=31;
                        switch (gid) {
                            case -1:
                            case 1:          /* Pfeile : */
                                nb=blk+gid;
                                if (nb>=max) nb=0;
                                if (nb<0) nb=max-1;
                                break;
                            case 0:           /* Info :  */
                                men=info;
                                max=NUM_INFO;
                                blk=-1;
                                nb=0;
                                break;
                        }
                    }
                    break;
          }
     }
}


VOID *OpenLibrary();

OpenStuff()
{
int t;
     MakeImages();
     if (!(IntuitionBase=
          OpenLibrary("intuition.library",0L)))
           ex(0);
                     /* Anmerkung: Falls Intuition nicht
                        geöffnet werden konnte, kann auch
                        kein Requester gebracht werden! */


     if (!(GfxBase=
          OpenLibrary("graphics.library",0L)))
          Error(UBP"Wo ist Graphics ?"),ex(0);
                                      /* Graphics */

     if (!(win=OpenWindow(&windef)))
          Error(UBP"Hilfe! Kein Window!"),ex(0);
                                 /* und das Window */

     rp=win->RPort; /*RastPort holen und vorbereiten */
     SetDrMd(rp,JAM2);
     SetAPen(rp,1L);
     SetBPen(rp,0L);

     if (!(fh=Open("par:",MODE_NEWFILE)))
                    /* Drucker zum Test öffnen.      */
          Error(UBP"Drucker belegt ?"),ex(0);

          Close(fh);
                    /* Jetzt ist das 'parallel.device'
                       vorerst resident und muß nicht
                       mehr von der Sys.-Disk geladen
                       werden...                     */

                    /*      Text-Längen eintragen :  */
     for (t=0;t<(NUM_MENS*6);t++)
          menu[t].Len=(BYTE)strlen(menu[t].Name);
     for (t=0;t<(NUM_INFO*6);t++)
          info[t].Len=(BYTE)strlen(info[t].Name);
}

ex(v)
int v;
/* Schließen aller Libraries und des Windows,
   Verlassen des Programms                          */
/* Wenn  v=0,  dann ohne Requester, ansonsten mit.  */
{
 if (v) v=Error(UBP"Programm verlassen ?");
 if (!v) {
      if (win)    CloseWindow(win);
      if (GfxBase) CloseLibrary(GfxBase);
      if (IntuitionBase) CloseLibrary(IntuitionBase);
      RemoveImages();
      Exit(0L);
     }
}

DoIt(num)
int num;
{
     switch (men[num].Type) {

      case pr_COMMAND:
                    /* Es   wurde  ein  Drucker-Befehl
                       angewählt                     */
       if (!(fh=Open("par:",MODE_NEWFILE)))
          Error("Drucker belegt ?"),ex(0);
       Write(fh,&(men[num].Prt),(long)(men[num].PLen));
       Write(fh,Beep,1L);
       Close(fh);
       break;

      case pr_EXTERN:
                    /* Es wurde ein Extern-Befehl ange-
                       wählt. Hier wird je nach Nummer
```

Listing: Printer Control Window V2.0

```
                            (in PLen) dekodiert, welche
                            Funktion ausgewählt wurde.     */
                switch (men[num].PLen) {
                    case 1:                         /* Quit */
                        ex(1);
                        break;
                    case 2:                         /* Info */
                        men=info;
                        max=NUM_INFO;
                        blk=-1;
                        nb=0;
                        break;
                    case 3:                  /*  norm. Menü */
                        men=menu;
                        max=NUM_MENS;
                        blk=-1;
                        nb=0;
                        break;
                }

      }
}

/* Diese Routine zeigt die aktuelle Menüseite an.Dabei
   wird vorher der Ausgabebereich gelöscht.             */

Display_em()
{
int t,b,n;
char p[MAXLINE+2];
  rp=win->RPort;
  b=blk*6;
  SetAPen(rp,0L);
  RectFill(rp,9L,14L,173L,83L);
  for (t=0;t<6;t++) {
   for (n=0;n<=MAXLINE;p[n++]=' ');
   strcpy(p,men[b+t].Name);
   SetAPen(rp,(long)(men[b+t].FPen));
   SetBPen(rp,(long)(men[b+t].BPen));
   Move(rp,12L,(long)((t*12)+21));
   Text(rp,men[b+t].Name,(long)strlen(men[b+t].Name));
  }
}


/* Dieser Teil erzeugt einen AutoRequest mit dem Text,
   den man beim Aufruf übergeben hat.                   */

struct IntuiText ITex3  = {0,1,JAM2,20,26,&TOPAZ80,
     NL,NL};          /*  wird vom Programm eingesetzt */
struct IntuiText ITex2  = {0,1,JAM2,20,13,&TOPAZ80,
     UBP"   Bitte bestätigen :",&ITex3};
struct IntuiText ITex1  = {0,1,JAM2,20,5,&TOPAZ80,
     UBP"   A C H T U N G  !á!",&ITex2};
struct IntuiText OKt    =
     {0,1,JAM2,6,3,&TOPAZ80,UBP"OK",NL};
struct IntuiText NEINt    =
     {0,1,JAM2,6,3,&TOPAZ80,UBP"Nein",NL};

int Error(txt)
UBYTE *txt;
{
     ITex3.IText=txt;
     return(AutoRequest(win,&ITex1,&NEINt,&OKt,
                        NL,NL,272L,69L));
}


MakeImages()
/* Kopiert Images ins Chip-Ram und biegt Zeiger um   */
{
     ID1=AllocMem(28L,MEMF_CHIP);
     ID2=AllocMem(28L,MEMF_CHIP);
     ID3=AllocMem(28L,MEMF_CHIP);
     if (!(ID4=AllocMem(28L,MEMF_CHIP)))
          Error("Nicht genug Chip-Memory"),
          Error("Gadgets können def. sein!");
                    /* Eigentlich   müßte   bei  jedem
                     AllocMem() der erfolgreiche Ab-
                     schluß   überprüft   werden,   aber
                     wenn der letzte OK ist, müßte der
                     auch   in   Ordnung sein ( in 99.9%
                     der Fälle...)                     */

     CopyIm(&ImageData2,ID2);
     CopyIm(&ImageData3,ID3);

     CopyIm(&ImageData4,ID4);
     CopyIm(&ImageData1,ID1);
     Image1.ImageData=ID1;
     Image2.ImageData=ID2;
     Image3.ImageData=ID3;
     Image4.ImageData=ID4;
}

RemoveImages()
{

/* Gibt den belegten Speicher an das System zurück.
   Hier sollte man auf jeden Fall prüfen, ob der Spei-
   cher  überhaupt geholt wurde, ansonsten bekommt man
   einen netten kleinen Guru... (für IDn=NULL!)       */

     if (ID1) FreeMem(ID1,28L);
     if (ID2) FreeMem(ID2,28L);
     if (ID3) FreeMem(ID3,28L);
     if (ID4) FreeMem(ID4,28L);
}

CopyIm(src,dest)
UBYTE *src,*dest;
{
int t;
     for (t=0;t<28;t++) *dest++=*src++;
}

WaitIT(n)
int n;
{

/*           Wartet auf n Intuiticks.                */

struct IntuiMessage *msg,*GetMsg();
ULONG IDCMP_old,cl;
     IDCMP_old=win->IDCMPFlags;
                           /*    Alte Flags sichern */
     ModifyIDCMP(win,IDCMP_old|INTUITICKS);
                           /*  und neue aktivieren. */
     for (;n;) {
          WaitPort(win->UserPort);
          msg=GetMsg(win->UserPort);
          cl=msg->Class;
          ReplyMsg(msg);
          if (cl==INTUITICKS) n--;
     }
     ModifyIDCMP(win,IDCMP_old);
                           /*    Alte Flags holen     */
     for (;msg;) {
                    /*   überflüssige Msgs abholen    */
          msg=GetMsg(win->UserPort);
          if (msg) ReplyMsg(msg);
     }
}
/*************** Ende von prcw.c *******************/
```

Listing: Printer Control Window V2.0

```
/*****************************************************/
/*                    pcw_defs.h                     */
/*         Include-File für Printer Control Window   */
/*          (c) 1990 Garry Glendown & AMIGA DOS      */
/*         Sprache : Aztec C                         */
/*****************************************************/

#define NL NULL

SHORT BorderVectors1[] = {0,0,195,0,195,4,0,4,0,0};
SHORT BorderVectors2[] = {0,0,18,0,18,38,0,38,0,0};
struct Border Border1 =
     {-2,-1,3,0,JAM1,5,BorderVectors1,NL};
struct Border Border2 =
     {-2,-1,3,0,JAM1,5,BorderVectors2,NL};

struct TextAttr TOPAZ80 =
     {(STRPTR)"topaz.font",TOPAZ_EIGHTY,0,0};
struct IntuiText IText4 =
     {1,0,JAM2,4,28,&TOPAZ80,(UBYTE *)"O",NL};
struct IntuiText IText3 =
     {1,0,JAM2,4,19,&TOPAZ80,(UBYTE *)"F",&IText4};
struct IntuiText IText2 =
     {1,0,JAM2,4,10,&TOPAZ80,(UBYTE *)"N",&IText3};
struct IntuiText IText1 =
     {1,0,JAM2,4, 1,&TOPAZ80,(UBYTE *)"I",&IText2};

USHORT ImageData1[] =
     {0xF83E,0xFBBE,0xFBBE,0x8BA2,0x739C,0xBBBA,0xDFF6,
     0xEFEE,0xF7DE,0xFBBE,0xFD7E,0xFEFE,0xFFFE,0xFFFE};
USHORT ImageData2[] =
     {0xFBBE,0xF93E,0xF83E,0x8BA2,0x0380,0x0380,0x739C,
     0xBBBA,0xDFF6,0xEFEE,0xF7DE,0xFBBE,0xFD7E,0xFEFE};
     USHORT ImageData3[] =
     {0xFFFE,0xFFFE,0xFEFE,0xFD7E,0xFBBE,0xF7DE,0xEFEE,
     0xDFF6,0xBBBA,0x739C,0x8BA2,0xFBBE,0xFBBE,0xF83E};
USHORT ImageData4[] =
     {0xFEFE,0xFD7E,0xFBBE,0xF7DE,0xEFEE,0xDFF6,0xBBBA,
     0x739C,0x0380,0x0380,0x8BA2,0xF83E,0xF93E,0xFBBE};

USHORT *ID1,*ID2,*ID3,*ID4,*AllocMem();

struct Image Image1 = {0,0,15,14,1,ImageData1,1,0,NL};
struct Image Image2 = {0,0,15,14,1,ImageData2,1,0,NL};
struct Image Image3 = {0,0,15,14,1,ImageData3,1,0,NL};
struct Image Image4 = {0,0,15,14,1,ImageData4,1,0,NL};

struct Gadget Gadget10 =
     {NL,9,86,192,3,GADGHBOX|GADGHIMAGE,RELVERIFY,
     BOOLGADGET,(APTR)&Border1,NL,NL,NL,NL,40,NL};
struct Gadget Gadget9 =
     {NL/*&Gadget10*/,188,30,15,37,NL,RELVERIFY,BOOLGADGET,
     (APTR)&Border2,NL,&IText1,NL,NL,31,NL};
struct Gadget Gadget8 =
     {&Gadget9,9,74,173,9,NL,RELVERIFY,BOOLGADGET,
     NL,NL,NL,NL,NL,26,NL};
struct Gadget Gadget7 =
     {&Gadget8,9,62,173,9,NL,RELVERIFY,BOOLGADGET,
     NL,NL,NL,NL,NL,25,NL};
struct Gadget Gadget6 =
     {&Gadget7,9,50,173,9,NL,RELVERIFY,BOOLGADGET,
     NL,NL,NL,NL,NL,24,NL};
struct Gadget Gadget5 =
     {&Gadget6,9,38,173,9,NL,RELVERIFY,BOOLGADGET,
     NL,NL,NL,NL,NL,23,NL};

struct Gadget Gadget4 =
     {&Gadget5,9,26,173,9,NL,RELVERIFY,BOOLGADGET,
     NL,NL,NL,NL,NL,22,NL};
struct Gadget Gadget3 =
     {&Gadget4,9,14,173,9,NL,RELVERIFY,BOOLGADGET,
     NL,NL,NL,NL,NL,21,NL};
struct Gadget Gadget2 =
     {&Gadget3,188,69,15,14,GADGHIMAGE|GADGIMAGE,RELVERIFY,
     BOOLGADGET,(APTR)&Image1,(APTR)&Image2,NL,NL,NL,32,NL};
struct Gadget Gadget1 =
     {&Gadget2,188,14,15,14,GADGHIMAGE|GADGIMAGE,RELVERIFY,
     BOOLGADGET,(APTR)&Image3,(APTR)&Image4,NL,NL,NL,30,NL};


struct NewWindow windef = {233,57,207,88,0,1,
     GADGETUP|ACTIVEWINDOW|INACTIVEWINDOW|RAWKEY,
     WINDOWDRAG|WINDOWDEPTH|RMBTRAP,&Gadget1,NL,
     (UBYTE *)"PCW by G.Glendown",NL,NL,0,0,0,0,
     WBENCHSCREEN};

/************ Ende von pcw_defs.h *******************/
```

Listing: Printer Control Window V2.0

Edgar Meyzis

Aufsteigerwissen Assembler

In der (zunächst?) letzten Folge unseres Assemblerkurses wollen wir auf die Hardware-Ebene hinabsteigen, um Zugang zu einem Bereich zu gewinnen, für den Assembler am besten geeignet ist. Wir werden einzelne Bildpunkte setzen, daraus eine Linie komponieren und schließlich Herrn Blitter "en personne" auftreten lassen.

Kursfahrplan

Teil 1. Einführung und Grundlagen

Teil 2. Vertiefung der Programmiertechnik

Teil 3. Praktische Arbeit mit Graphics

Teil 4. Praktische Arbeit mit DOS

Teil 5. Hardwarenahe Programmierung von Grafik

Unser Rückblick soll der vierten Folge gelten, in der wir Ihnen die Aufgabe stellten, zwei ähnliche Routinen zusammenzufassen. Die Lösung kann sehr unterschiedlich erfolgen.

Ein Blick zurück

Getreu unserem Kursmotto haben wir einen einfachen, leicht nachvollziehbaren Weg gewählt, wie in Listing 1 und 2 dargestellt. Wir kopieren unsere Quelldatei "O4.asm" (Kurzform für ProjektMenu.asm) nach "O5.asm", entfernen daraus die Routinen "OpenBildDatei", "CloseBildDatei" sowie "BildSchreiben" und bauen "BildLesen" entsprechend Listing 1 aus. Weiterhin ist die Routine "Menu0Auswerten" entsprechend Listing 2 zu ändern.

Die Arbeitsweise unserer neuen Routine "BildTransferieren" steuern wir mit einem Parameter, den wir in Register d2 übergeben. Alternativ hätten wir auch über den Stack gehen können. Das Register d2 wurde gewählt, weil es gleich beim Öffnen der Datei als Parameterregister ohnehin gefordert ist. Es bot sich an, die Konstanten "modeOldFile" und "modeNewFile" zur Steuerung von "BildTransferieren" zu verwenden.

Die neue Routine ist gegenüber dem bisherigen Vorgehen robuster geworden. Sie berücksichtigt, daß die Schreib- und Leseroutinen des "DOS" nicht immer ungestört ablaufen (Beispiel: Diskette ist voll beschrieben). "DOS" gibt im Fehlerfall einen negativen Wert (in d0) zurück, auf den mit "bmi.s \e" (branch minus) reagiert wird. Durch eine ausführliche Fehlerbehandlung einschließlich einer Ausgabe der Ursache könnte die Routine verbessert werden. In der vorliegenden Form wird die Datei geschlossen (sofern vorher geöffnet) und so getan, als sei nichts geschehen. Das Vorgehen bewahrt vor "Abstürzen", ist aber benutzerunfreundlich.

Es soll noch auf den Assemblerbefehl "movem.l" eingegangen werden, den wir häufig eingesetzt haben, ohne zu betrachten, in welcher Reihenfolge die Register auf dem Stack abgelegt bzw. von dort geladen werden. Um es kurz zu machen, die Reihenfolge der Register im Quelltext ist bedeutungslos. Der Assembler wertet unsere Notation aus und setzt im zweiten Wort des Maschinenbefehls, das die Registermaske bildet, für jedes angesprochene Register ein bestimmtes Bit (ein Wort = 16 Bits = 8 Adreß- und 8 Datenregister der CPU). Die Registermaske bestimmt somit die Reihenfolge der Register auf dem Stack. Diese Feststellung läßt sich verallgemeinern: Mit der "movem-Instruktion" werden Registerinhalte so im Arbeitsspeicher (also auch auf dem Stack) abgelegt, daß an der untersten Adresse die Datenregister, mit d0 beginnend, und anschließend die Adreßregister (a0 zuerst) Platz für ihren Inhalt finden. Mit zunehmender Speicheradresse steigt somit auch die Registernummer, beginnend bei d0 und endend bei a7. Im Quelltext nicht benannte Register werden einfach übergangen.

Bei einem Stack ist zu bedenken, daß er von oben nach unten wächst. Für den konkreten Fall unserer Routine "BildTransferieren" ergibt sich: Unabhängig von der Reihenfolge der Nennung der Register in der Anweisung "movem.l a2/d7/d2,-(sp)" weist "sp" (Stack Pointer) auf den Anfang der Speicherstellen, die den Inhalt von d2 aufgenommen haben.

Ohne in die Arbeitsweise des Blitters tief einzusteigen (das überlassen wir unserer "Blitter-Ecke", die sich des Themas seit Ausgabe 2/90 annimmt), wollen wir nun "hardwarenäher" programmieren, um Grafikausgaben zu beschleunigen. Einen krönenden Abschluß muß eine strapaziöse Wüstendurchquerung schließlich finden. Der Kreis schließt sich:

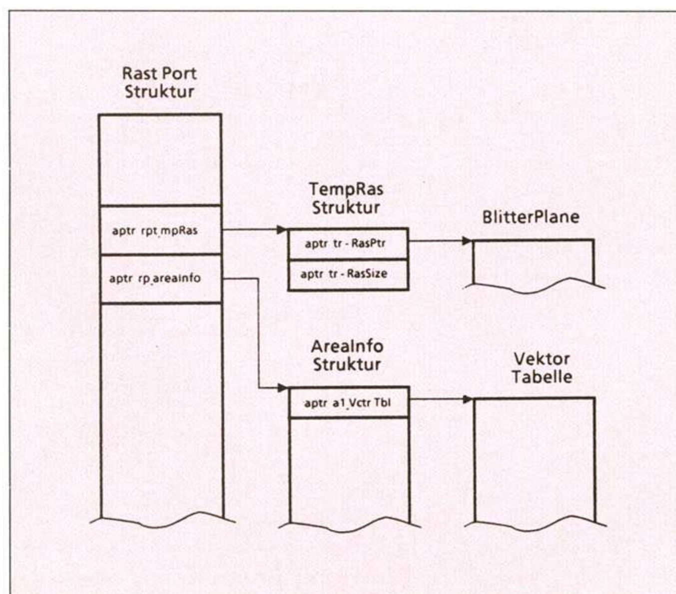


Bild 1. Die Routine "AreaOperation Vorbereiten" unterstützt unseren "Rastport" mit benötigten Datenstrukturen

Fülle Ei mit Dampf

Heute wollen wir das Ei mit Dampf füllen, ich meine die Abarbeitung der Routine "FuelleEi" beschleunigen. Mit Rücksicht auf die hinzugekommenen Leser ist "FuelleEi" in Listing 3 wiedergegeben, allerdings in überarbeiteter Form, da wir zwischenzeitlich unsere Programmiertechnik verfeinert haben. Erstellen Sie anhand des Listings die Textdatei "G5.asm". Wir werden damit weiterarbeiten, um einige Funktionen des Blitters zu demonstrieren, die Sie bei Bedarf in Ihre "PaintBox" übernehmen können. Die Routine "AreaBlitterDemo", die sich zunächst nur um "FuelleEi" kümmert, rufen wir menügesteuert aus unserem Hauptprogramm "P5" auf. Sie können dafür ein weiteres Menü oder einen Menüpunkt anlegen. Die Technik dürfte Ihnen mittlerweile geläufig sein. Wir hingegen erweitern in "Z5" (vormals "Z4") die Routine "ZeichneBox" um den Aufruf "bsr AreaBlitterDemo". Noch haben wir nicht an "speed" zugelegt. Wir wollen zunächst nur die Gewißheit gewinnen, daß die Routine fehlerfrei arbeitet, um einen sicheren Ansatzpunkt zu schaffen.

So, nun kommt Dampf ins Ei, wie Sie Listing 4 entnehmen können. Bei dem bisherigen Vorgehen haben wir dem Blitter die Arbeitsbedingungen vorenthalten, unter denen er erst richtig zupacken kann. Da fehlt uns zunächst die

Spielwiese für den Blitter, die Area, die wir mit der Systemroutine "AllocRaster" in der Größe unseres Bildschirms (640 * 256) (Routine "BlitterPlaneAnlegen") schaffen. Weiterhin ist eine Kurzbeschreibung dieser "Blitter-Plane" mit der Systemroutine "InitTempRas" in einen Zwischenspeicher (Routine "TempRasAnlegen") zu schreiben und dessen Adresse in der Rast-Port-Struktur zu vermerken. Bild 1 stellt die Zusammenhänge dar. Angeforderter Arbeitsspeicher ist an das Betriebssystem zurückzumelden (mit "FreeRaster"), wenn nicht mehr benötigt. Das erledigt "BlitterPlaneAufgeben".
Wir packen das "Ei" quasi in ein Sandwich, bestehend aus den Routinen "AreaOperationenVorbereiten" und "AreaOperationenAbschliessen", um endlich in den vollen Genuß des Blitters zu gelangen. Ergänzen Sie bitte Ihr Listing 3 durch Listing 4. Na, wie gefällt Ihnen das? Sie können das Makro "FlaecheFuell" nun auch mit dem Füllmodus "1" aufrufen, um die Möglichkeit zu nutzen, beliebig begrenzte Flächen zu füllen und nicht einfach alles zu übertünchen, was innerhalb des Bereichs liegt, der mit dem "AreaOutLinePen" markiert ist. Die Ellipse ist fast gleichberechtigt. Aber es kommt noch besser! In Listing 4 haben wir bereits zwei "leere" Routinen ("AreaInfoAnlegen" und "AreaInfoAufgeben") vorgesehen, die wir im nächsten Abschnitt mit Code füllen werden.
Aus den Makros "FlaecheFuell" und den "AreaOperation"-Routinen könnten Sie Ihrer "PaintBox" ein Geschenk entwickeln, um beliebige Flächen zu füllen. Der Füllmodus sollte per Menü wählbar sein, ebenso die Funktion "Füllen". Letztere sollte die Position der Maus feststellen, wenn Sie den linken Knopf drücken, und die Fläche in der Farbe des Vordergrundstiftes (APen) füllen. Als Muster für das Vorgehen könnte die bewährte Routine "FreeDraw" dienen.

Noch fetziger und mit Mustern

Bisher haben wir nur die "Flood-Funktion" der "Blitterorientierten" Systemroutinen genutzt. Nun gehen wir einen Schritt weiter und lassen den Blitter für uns geometrische Gebilde zeichnen sowie die dabei entstehenden Flächen füllen. Dazu müssen wir noch eine "AreaInfo-Struktur" gemäß Listing 5 anlegen und deren Adresse in die Rast-Port-Struktur eintragen. Weiterhin ist es erforderlich, für die zu zeichnenden Figuren eine Tabelle anzulegen, in die die Routine "AreaDraw" Zeichenvektoren für den Blitter vermerkt. Der Zusammenhang zwischen den Strukturen "RastPort", TempRas", "AreaInfo" und der Vektorentabelle geht aus Bild 1 hervor.
Reservierte Speicherbereiche sind nach Abschluß der "Area-Operationen" an das Betriebssystem zurückzumelden. Den angelegten Strukturen können die jeweiligen Adressen entnommen werden, so daß es überflüssig ist, globale Variablen einzurichten. Auf die Reihenfolge der Rückmeldungen ist zu achten, um nicht die Gurus hinter der nächsten Sanddüne herauszufordern. "AreaInfoAufgeben" ist eine der Routinen, die Speicherplatz zurückmeldet. Sie wäre robuster, wenn sie zunächst prüfen würde, ob die Adressen der Speicherbereiche überhaupt gültig sind.
Mit "AreaInfoAnlegen" und "AreaInfoAufgeben" haben wir nun alle Voraussetzungen geschaffen, mit "Areas", das heißt mit dem Blitter, zu arbeiten. Uns steht nun die uneingeschränkte Anwendung der Systemroutinen

– AreaMove (Startpunkt eines Vielecks bestimmen),
– AreaDraw (Eckpunkte des Polygons festlegen),
– AreaEllipse (Ellipse definieren),
– AreaEnd (mit zuvorgenannten Befehlen bestimmte Figuren nun endlich zeichnen und ausfüllen).

offen. Die Sprungvektoren sind in Listing 5 vermerkt. Eine Demonstration der neuen Befehle bietet die Routine "Polygon-UndEllipse".
Beliebige Flächen können wir nun einfarbig füllen. Die Polygontechnik ließe sich einfach in die PaintBox integrieren. Auf unserer letzten Wüstenetappe wollen wir nun ein paar Spuren im Sand hinterlassen (besser als Cola-Dosen), das heißt Flächen mit Mustern füllen. Auch das ist mit dem Blitter leicht möglich.
Listing 6 enthält alle Informationen über die Verwendung einfarbiger Muster. Das Makro "SetAfPt" (Set Area Fill Pattern) meldet das Muster bei der Rast-Port-Struktur an. Von nun an wird mit dem Muster gefüllt. Wir haben "Muster0" global deklariert. Muster haben stets eine Breite von 16 Bits, begrenzt durch die Auslegung des Blitters. Bei breiteren Flächen wiederholt sich das Muster. Die Höhe ist beliebig wählbar, jedoch nur in Schritten von Zweierpotenzen. Als Höhe ist nicht die Anzahl der Wörter, sondern der Exponent der Zweierpotenz anzugeben (in unserem Fall bei acht Wörtern somit drei).
Es lassen sich auch mehrfarbige Muster einfach erzeugen. Für jede Bitplane ist dann ein Muster anzulegen. Die Höhe des Musters ist wie im ersten Fall für eine Bitplane anzugeben, jedoch mit negativem Vorzeichen. Ergänzen Sie Listing 6 entsprechend. Sie werden von den Möglichkeiten begeistert sein.

Superschnelle Punkte

Mit dem Blitter haben wir schon sehr nahe an der Hardware gearbeitet.

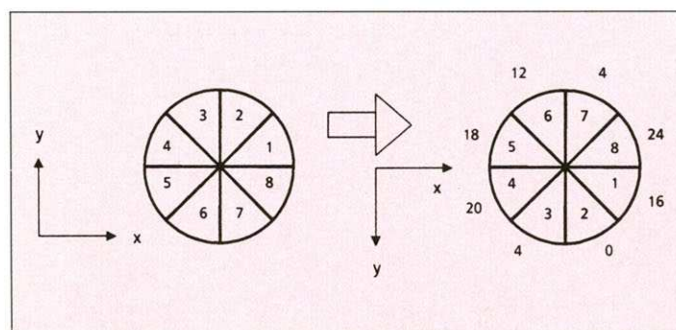


Bild 2. Durch Drehen um die x-Achse passen wir die Oktantennumerierung der Lage des Koordinatensystems der Amiga-Grafik an. Außerhalb des rechten Kreises: Oktantenbezeichnung nach Blitterart

Mit Listing 7 stellen wir Ihnen Routinen vor, die Punkte setzen. Betrachten wir zunächst "SetzePunkt": Als erster Parameter ist die Adresse der Bitmap-Struktur anzugeben, auf die in der Rast-Port-Struktur verwiesen wird. In "Graphics.i" finden Sie die Vereinbarung der Bitmap-Struktur. Sie gibt Auskunft über die Größe und die Tiefe (das heißt Anzahl der Bitplanes) eines RastPort. Weiterhin enthält Sie ab "bm_planes" ein Feld mit den Adressen der einzelnen Bitplanes. Die Adressen benötigen wir, um in den Planes Punkte zu setzen. Als Parameter für "SetzePunkt" wäre der aktuelle RastPort gleichfalls geeignet.
Gehen wir nun die Anweisungen von "SetzePunkt" durch. Zunächst holen wir uns die Adresse der ersten Bitplane. Dann ermitteln wir, in welchem Byte einer Bildschirmzeile die X-Position liegt. Die Multiplikation der Y-Position mit 80 (640 Bildpunkte geteilt durch acht) ergibt die Adresse der Zeile, bezogen auf den Anfang der Bitplane. (Die Multiplikation erfordert mehr Zeit als ein "shift #6,.." und ein "shift #4,.." sowie Addition der erhaltenen Werte. Aus Gründen der Anschaulichkeit haben wir auf diese Möglichkeit der Optimierung verzichtet.) Die Addition der Zeilenadresse mit der Adresse des Bytes innerhalb der Zeile ergibt schließlich die Adresse des Bytes, bezogen auf den jeweiligen Anfang unserer vier Bitplanes, in dem wir ein Bit, das heißt unseren Punkt, zu setzen haben.
Die Adresse des Punktes innerhalb des Bytes erhalten wir durch die drei Anweisungen "and.w #7,d3 sub.q #7 und neg.w d3". Bei der ganzzahligen Multiplikation durch acht haben wir die unteren drei Bits verloren. Die holen wir uns mit der ersten Anweisung wieder und kennen nun die Position des zu setzenden Pixels innerhalb des Bytes. Leider ist noch eine Umformung des erhaltenen Ergebnisses notwendig, weil Bits von rechts nach links durchnumeriert werden, während die X-Positionen von links nach rechts zunehmen. Der aufgezeigten Gegenläufigkeit wird durch die beiden weiteren Anweisungen Rechnung getragen. Das Register d3 enthält schließlich die genaue Bezeichnung des Bits.

Unmittelbar hinter dem Label "\L0" testen wir, ob in der aktuellen Bitplane ein Bit zu setzen ist. Die Bits null bis drei entsprechen unseren vier Bitplanes. Falls kein Bit zu setzen ist, verzweigen wir zu "\L1" und holen uns die Adresse der nächsten Bitplane. Dann schieben im Farbregister d2 die Bits nach rechts, um das mit der aktuellen Bitplane korrespondierende Bit an der Position null zu haben. Schließlich fangen wir wieder bei "\L0" an. Dieses Vorgehen ist sehr schnell. Es wird jedoch von dem Nachteil begleitet, daß es zu Farbverfälschungen kommen kann, weil nur Punkte hinzugefügt werden.

Wenn ein Punkt zu setzen ist, dann wird zu der Adresse der Bitplane (a1) die Adresse innerhalb der Plane (d1) addiert und das Bit gesetzt ("bset d3,(a1)"). Die Routine "SetzePunkt" hat sich in Simulationssystemen bestens bewährt.

In der "SetzePunktDemo" wird eine Gerade gezogen, deren Farbe von Punkt zu Punkt wechselt. Für jeden einzelnen Punkt der Gerade werden die Koordinaten und die Farbe an "SetzePunkt" übergeben. Ein ungeheuerer Aufwand für eine Gerade. Im Fall einer Kurve sieht das aber schon ganz anders aus. Wie gefällt Ihnen die Geschwindigkeit der Routine?

Linien mit dem Blitter

Einzelne Punkte können wir nun setzen, und auch für geschwungene Linien haben wir einen rasanten Weg gefunden. Nun fordern wir den Blitter unmittelbar auf, für uns Geraden zu ziehen. Es gilt zunächst den Oktanten (Achtelkreis) zu bestimmen, der die grobe Richtung der Linie beschreibt. Wo fängt man bei der Bezeichnung der Oktanten bei einem Kreis an? Konventionen besagen, daß bei "drei Uhr" der Oktant Null beginnt und daß entgegengesetzt dem Uhrzeigersinn durchnumeriert wird. Auf dem Amiga ist aber alles anders. Aus Gründen der Anschaulichkeit beugen wir uns der hier üblichen Konvention, die Oktanten von eins bis acht durchzunumerieren. Am Anfang des Listing 8 finden Sie entsprechende Vereinbarungen, die in die Datei "incl/Hardware.i" eingehen sollten.

Wir könnten somit bei einer Bezeichnung der Oktanten gemäß der linken Hälfte von Bild 2 bleiben. Im Gegensatz zu den mathematischen Konventionen liegt beim Amiga der Nullpunkt links oben in der Ecke und nicht links unten. Es folgt: Ein zunehmendes Delta-Y (Y2 – Y1) bedeutet auf dem Amiga keine positive, sondern eine negative Steigung. Aus Gründen der Anschaulichkeit ist es daher zweckmäßig, die Oktantenbezeichnung entsprechend Bild 2, rechter Teil, der Lage des Koordinatensystems des Amiga anzupassen.

Leider versteht der Blitter nicht unsere Numerierung der Oktanten. Er will anders informiert werden. Dazu dienen die Dezimalzahlen außerhalb des rechten Kreises (Bild 2). Sie finden die Werte auch in Listing 8 in der Tabelle "Oktanten" wieder. Die Berechnung von Oktanten ist wesentlich einfacher als häufig angenommen. In Bild 2 werden die notwendigen Überlegungen ausführlich dargestellt.

Mit Blick auf Bild 2, rechter Teil, stellen wir fest, daß der Endpunkt der Linie in den Oktanten 7, 8, 1 oder 2 liegen muß, wenn X1 kleiner X2 gilt (X1 > X2 : Oktanten 3, 4, 5 oder 6).

Weiterhin gilt: Wenn Y1 kleiner Y2 ist, dann liegt der Endpunkt in den Oktanten 5, 6, 7 oder 8 (Y1 > Y2 : Oktanten 4, 3, 2 oder 1).

Mit den bisherigen Folgerungen könnte man nur auf den Viertelkreis schließen, in dem der Endpunkt der Linie liegt. Eine weitere Bedingung hilft uns, vom Viertelkreis zum Oktanten zu gelangen, indem wir die Steigung der Linie einbeziehen: Wenn der Absolutwert von Delta-X größer ist als der Absolutwert von Delta-Y, dann kommen nur die Oktanten 5, 8, 4 oder 1 in Frage; im umgekehrten Fall verbleiben die Oktanten 6, 7, 3 oder 2.

Durch Auswertung der drei Bedingungen läßt sich der gesuchte Oktant sicher bestimmen. Dazu könnte man zum Beispiel entsprechend Listing 10 verfahren. Wer mag schon die ineinander verwobenen Verzweigungen? Einem derartig formulierten Algorithmus mangelt es an Klarheit. An der Aussage ändert sich nicht viel, wenn man sicherstellt, daß X1 stets größer als X2 ist. Durch Vertauschen ließe sich das bewerkstelligen, und der Blitter hätte auch nichts dagegen. Die möglichen Lösungen würden dann immer im rechten Halbkreis liegen.

Wir gehen einen etwas anderen Weg und wenden in Listing 8 ein Stück Mengenlehre an, das wir in Listing 9 noch ein wenig verfeinern. Dazu legen wir uns eine Tabelle mit den Oktanten an. Vergleichen Sie bitte die drei formulierten Bedingungen mit der Tabelle, die mit dem Label "X1_kl_X2" beginnt. Erkennen Sie die Abbildung der hergeleiteten Bedingungen. Wir wollen entsprechend der Ergebnisse der drei Vergleiche die drei Zeilen der Tabelle bestimmen, die den gesuchten Oktanten enthalten. Wir wissen, daß der gesuchte Oktant in allen drei Zeilen vorkommt. Jede Zeile ist 32 Bits lang. Folglich muß eine UND-Verknüpfung (Schnittmenge) der drei Zeilen den gesuchten Oktanten zum Ergebnis haben. Genau diese Aufgabe leistet der erste Teil der Routine (Listing 8) bis zum Label "\Linie".

Warum muß eigentlich ein Langwort mit acht Halbbytes (Nibbles) herhalten, um die Oktanten aufzunehmen? Ein Byte müßte doch als Datenstruktur ausreichen, da es gleichfalls acht verschiedene Werte annehmen kann? Sehr richtig! In Listing 9 führen wir auch gleich diese Verfeinerung ein. Der Code wird insgesamt kürzer und im Schnitt auch schneller.

Verfolgen wir nun Listing 8 ab "\Linie" weiter, nachdem uns der erste Teil so beschäftigt hat: Wir haben genauso wie für "SetzePunkt" die Adresse des Bytes zu bestimmen, in dem der Bildpunkt liegt. Auch die Position des Pixel innerhalb des Bytes ist gleichermaßen zu bestimmen. Am Schluß der Routine, vor dem Label "\wart" treffen wir auch auf die uns schon geläufige Technik, die Adresse der nächsten Bitplane zu erlangen.

Auf die eigentliche Versorgung des Blitter mit Daten wollen wir hier nicht eingehen; das macht die "Blitter-Ecke". Abweichend zu "SetzePunkt" benutzen wir ein Zählregister, um die vier Bitplanes zu bearbeiten. Weiterhin löschen wir in einer Bitplane die Bits an der Stelle der Linie, die nicht zu ihrer Farbe passen, sie somit umfärben würden. Das Löschen hätten wir durch Ändern der Maske in "bltdat(a1)" auf Null erledigen können. Eleganter erscheint uns jedoch der Wechsel der "Minterms" von #$FA auf #$A, zumal die Möglichkeit erhalten bleibt, gemusterte Linien zu ziehen. In Bild 3 haben wir den Versuch unternommen darzustellen, wie die drei möglichen Signalquellen durch den Blitter im Linienmodus verarbeitet werden. Als Quellen dienen:

1. die blitterintern erzeugte Anweisung, ein Bit zu setzen (oder auch nicht), Kanal A,

2. die Maske mit dem Linienmuster, Kanal B, der wir schon früher begegnet sind und

3. der Inhalt der Bitplane, Kanal C, die zu bearbeiten ist.

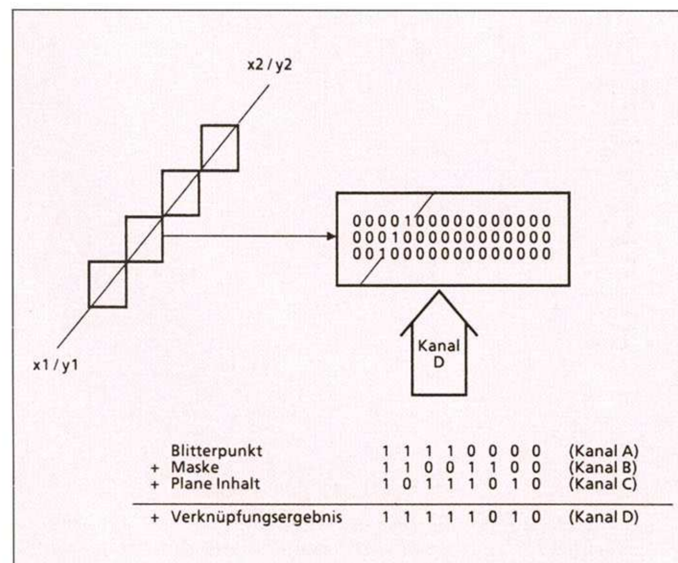


Bild 3. Der Blitter läßt den Drachen steigen

Die Bitplane ist somit Quelle (Kanal C) und Ziel (Kanal D) zugleich. Die MinTerms (kurze boolsche Ausdrücke) bestimmen, wie die Informationen der Kanäle zu verknüpfen sind.

Beispiel: Die zweite Spalte von links ist wie folgt zu interpretieren: Wenn der Blitter intern berechnet hat, daß ein Punkt zu setzen ist (Blitterpunkt = 1), und die Maske an der Bitposition des Punktes eine Eins enthält (Maske = 1) und an der entsprechenden Stelle der Bitplane kein Punkt gesetzt ist, dann soll der Blitter tun, was er meint, nämlich den Punkt setzen.

Beachten Sie bitte, daß Sie die Zeile mit den Verknüpfungsergebnissen selbst bestimmen. Durch Eintragen von Null oder Eins legen Sie fest, wie der Blitter zu arbeiten hat. Bedenken Sie bitte auch, daß der Blitter nicht punktweise, sondern zeilenweise arbeitet und somit das von Ihnen vorgegebene "Verknüpfungsverhalten) (Wortschöpfung!) jeweils für die Breite eines Wortes gilt. Mit der Vorgabe legen Sie eigentlich die Verknüpfungsregeln fest. Die Reihenfolge der Spalten ist so (unveränderlich) festgelegt, wie in Bild 3 dargestellt [1]. Das erwartete Verhalten des Blitters (Verknüpfungsergebnis) läßt sich sedezimal ausdrücken. Im dargestellten Beispiel ergibt sich #$FA.

Prüfen Sie es in Listing 8 nach: Wir füttern den Blitter mit dem Wert mit den Anweisungen "or.w #$FA,con0" (kurz nach \Li- nie) und "move.b #$FA, con0" (kurz nach "\warte"). Ist in der jeweiligen Bitplane kein Punkt zu setzen, dann beschränken wir uns auf #$A (unmittelbar vor \n). Vergleichen Sie unser Verhalten mit Bild 3. Im letzten Fall begnügen wir uns mit der Übernahme bereits gesetzter Bildpunkte, wie im rechten Teil des Bildes dargestellt. Sollten Sie dem Blitter bei der Arbeit zuschauen wollen, dann ersetzen Sie bitte #$FA vorübergehend durch #$40.

Ziel erreicht

Das Ende der heutigen Etappe bedeutet zugleich, daß wir das Ende des Kurses erreicht haben. Es hat Freude gemacht, für Sie zu schreiben. Teilen Sie uns mit, wenn Sie Interesse an einer ähnlichen Tour verspüren. Warum nicht einmal durch das Packeis der Arktis?

(mm)

Listings

```

BildTransferieren ; Parameter in d2
*-----------------   #modeOldFile : Bild einlesen
*                    #modeNewFile : Bild speichern

   movem.l a2/d7/d2,-(sp)        ; Arbreg + Param sich.

   move.l gadgBuffer(locals),d1 ; zu oeffnende Datei
   CallDos Open                 ; Mode bereits in d2
   move.l d0,bildHdl(locals)    ; FileHandle oder null
   beq.s \f                     ; falls nicht geklappt

   move.w #3,d7                 ; vier Bitplanes
   move.l Screen(globals),a2    ; Zeiger auf Screen

*  in Screen-Struktur ist BitMap-Struktur eingelagert,
die
*  ab bm_Planes, die Adressen unserer Bitplanes enthaelt

lea.l sc_bitMap+bm_planes(a2),a2      ; auf PlaneSpeich.
   move.l (a2),d2                     ; dieselbe holen
\speicherLoop

*  Jede Zeile enthaelt 640 Bildpunkte, d.h. 80 Bytes
*  Unser Screen ist 256 Bildpunkte hoch

   move.l #80*256,d3            ; Anzahl Bytes pro
Plane $5000
   move.l bildHdl(locals),d1 ; von Datei

   cmp.l  #modeNewFile,(sp) ; Bild speichern ?
   beq.s  \w                ; ja
   CallDos Read             ; 1 Plane einlesen
   bra.s \x                 ; zur Fehlerpruefung
\w CallDos Write         ; 1 Plane speichern
\x bmi.s \e              ; falls Fehler aufgetreten

   adda.l #4,a2          ; auf naechste Plane-
Speicher-Adr
   move.l (a2),d2        ; die Plane-Adr holen
   dbra d7,\speicherLoop ; und lesen

\e move.l bildHdl(locals),d1 ; FileHandle Bilddatei
   beq.s \f                  ; falls ungueltig
   CallDos Close
\f movem.l (sp)+,a2/d7/d2
   rts
```

Listing 1: Loesung (Teil 1) zur Aufgabe aus Folge 4

Durch Parametrisierung lassen sich die Schreib- und Leseroutinen leicht zusammenfassen

```

   .....

   tst.w   ItemNr(locals); Laden oder Sichern ?
   bne.s   \s
   move.l  #modeOldFile,d2 ; Parameteruebergabe in d2
   bra.s   \bt
\s move.l  #modeNewFile,d2 ; Sichern
\bt
   bsr BildTransferieren
\cl
   move.l (sp)+,a3

   .....
```

Listing 2: Teil 2 der Loesung zur Aufgabe aus der letzten Folge
Die Routine "Menu0Auswerten" ist wie dargestellt zu aendern, um "BildTransferieren" aufzurufen.

```
* Grafikspielereien (G5.asm)

  include "incl/Types.i"
  include "incl/Graphics.i"
  include "incl/Intuition.i"

AllocRaster = -492 ; (width,height)(d0/d1)
FreeRaster  = -498 ; (planePtr,width,height)(a0,d0/d1)
InitTempRas = -468 ; (tmpras,buff,size)(a0/a1,d0)

* Export

 PUBLIC AreaBlitterDemo ; ausserhalb verfuegbar machen

* Import

  PUBLIC GraphBase ; aus Hauptmodul
  PUBLIC RastPort

* Registergebrauch

globals  equr a5   ; Zeiger auf globale Variablen
our_Win  equr a4   ; Zeiger auf geoeffnetes Fenster



DoRpFunct macro                   ; Grafikfunktion in einem
*--------                        Rastport (Rp) ausfuehren
*                                Param: Funktion
  move.l RastPort(globals),a1 ; Rastport-Adr holen
  CallGraph \1              ; Funktion ausfuehren
 endm


SetzePen  macro
*------                         Param: XPos, YPos
  move.w \1,d0             ; XPos in Parameterreg.
  move.w \2,d1             ; YPos in Parameterreg.
  DoRpFunct Move ; Stiftposition setzen
 endm

FaerbeAPen macro
*---------             Para:  Farbregisternummer
  move.w \1,d0         ; Farbe ins Parameterregister
  DoRpFunct SetAPen
 endm

SetOpen macro
*------                              Para: Farbnummer
  move.l RastPort(globals),a1  ; Rastport-Adresse
  move.b \1,rp_aOlPen(a1)      ; Outlinefarbe
in Rp aendern
 endm


Ellipse macro
*------                    Para: XM, YM, Xr, Yr
  move.w \1,d0
  move.w \2,d1
  move.w \3,d2
  move.w \4,d3
  DoRpFunct DrawEllipse
 endm


FlaecheFuell macro


*-----------           Para: Mode, XPos, YPos
  move.w \1,d2        ; Fuell Mode, 0 oder 1
  move.w \2,d0        ; XPos
  move.w \3,d1        ; YPos
  DoRpFunct Flood
 endm


FuelleEi
*-------
  FaerbeAPen #13
  Ellipse #200,#110,#200,#100
  SetOpen #13
  FlaecheFuell #0,#200,#110
  rts

AreaBlitterDemo
*--------------
  bsr FuelleEi
  rts

  end
```

Listing 3: Ohne volle Nutzung des Blitters werden Flaechen sehr langsam gefuellt.

Listings zum Assemblerkurs

```
  include "incl/Types.i"
  include "incl/Graphics.i"
  include "incl/Intuition.i"
  include "incl/Exec.i"

AllocRaster = -492 ; (width,height)(d0/d1)
FreeRaster  = -498 ; (planePtr,width,height)(a0,d0/d1)
InitTempRas = -468 ; (tmpras,buff,size)(a0/a1,d0)

* TempRas
  struktur
   aptr tr_RasPtr
   aptr tr_RasSize
  struktLaenge tr_size


BlitterPlaneAnlegen
*------------------
  move.w wd_Width(our_Win),d0  ; Pixel/Zeile
  move.w wd_Height(our_Win),d1 ; Bildhoehe
  CallGraph AllocRaster        ; Adr BlitterPlane
  rts                          ; in d0


BlitterPlaneAufgeben           ; Parameter
*------------------              a0 : Adr BlitterPlane
  move.w wd_Width(our_Win),d0  ; Pixel/Zeile
  move.w wd_Height(our_Win),d1 ; Pixel/Spalte
  CallGraph FreeRaster
  rts


TempRasAnlegen ; Parameter: a1 Adr BlitterPlane
*-------------                     ; Ergebnis in d0
  move.l a2,-(sp)
  move.l a1,a2
  AllocMemory #tr_size,#Mem_Clear ; TempRas-Struktur
  tst.l d0                        ; angelegt ?
  beq.s \f                        ; falls nicht
  move.l d0,a0                    ; Adr TempRas-
Struktur
  move.l a0,-(sp)                 ; Adr TempRas merken
  clr.l  d0
  move.w wd_Width(our_Win),d0  ; Pixel/Zeile
  divu   #8,d0                 ; Byte/Zeile
  mulu   wd_Height(our_Win),d0 ; Byte/Plane
  move.l a2,a1
  CallGraph InitTempRas

  move.l wd_RPort(our_Win),a0 ; Rastport Adr holen und
  move.l (sp),rp_tmpRas(a0)   ; Adr TempRas eintragen
  move.l (sp)+,d0             ; Adr TempRas
\f
  move.l (sp)+,a2
  rts


TempRasAufgeben
*--------------
  move.l a2,-(sp)
  move.l RastPort(globals),a2 ; aus RastPort
  FreeMemory rp_tmpRas(a2),#tr_size ; tempRas aufgeben
  clr.l rp_tmpRas(a2)         ; in RastPort vermerken
  move.l (sp)+,a2
  rts



AreaInfoAnlegen
*--------------
  rts

AreaInfoAufgeben
*---------------
  rts


AreaOperationenVorbereiten ; ohne Parameter, Ergebnis
in d0
*------------------------   Ergebnis:Adr BlitterPl.
o. null
  move.l d2,-(sp)
  bsr BlitterPlaneAnlegen
  move.l d0,d2                ; Adr BlitterPlane merken
  beq.s \z                    ; keine BlitterPlane da

  bsr TempRasAnlegen
  tst.l d0                    ; angelegt + eingetragen ?
  beq.s \x                    ; nein

  move.l #20,d0               ; 20 Koordinatenpaare
  bsr AreaInfoAnlegen
  tst.l d0                    ; angelegt + eingetragen ?
  bne.s \z                    ; ja

  bsr TempRasAufgeben         ; da kein AreaInfo verfueg
bar
\x
  move.l d2,a0                  ; Adr BlitterPlane
  bsr BlitterPlaneAufgeben      ; Speicher zurueck
  clr.l (sp)                    ; Fehler signalisieren
\z
  move.l d2,d0                  ; Ergebnis holen
  move.l (sp)+,d2
  rts


AreaOperationenAbschliessen
*--------------------------
  move.l RastPort(globals),a0 ; aus RastPort
  move.l rp_tmpRas(a0),a0     ; ueber TempRas
  move.l tr_RasPtr(a0),a0     ; Adr BlitterPlane holen
  bsr BlitterPlaneAufgeben
  bsr TempRasAufgeben
  bsr AreaInfoAufgeben
  rts



FuelleEi
*-------
  FaerbeAPen #13              ; Fuellt Flaeche im Mod.0
  Ellipse #200,#110,#200,#60 ; mit einer Farbe
  SetOpen #13
  FlaecheFuell #0,#200,#110

Beispiel2                     ; Fuellt Flaeche im Mod. 1
  FaerbeAPen #9               ; mit Farbe 9
  Ellipse #200,#80,#70,#60
  FlaecheFuell #1,#200,#80
X
  FaerbeAPen #12
  FlaechePuell #1,#200,#30

  rts


AreaBlitterDemo
*--------------
  bsr AreaOperationenVorbereiten
  tst.l d0
  beq.s \f

  bsr FuelleEi

  bsr AreaOperationenAbschliessen
\f
  rts
```

Listing 4: Gebt dem Blitter seine Spielwiese, damit er Flaechen blitzschnell fuellen kann.

```

InitArea    = -282 ; (AreaInfo,vectorTable,vectorTable
Size)
*                    (a0/a1,d0)
AreaDraw    = -258 ; (RastPort,XPos,YPos)(A1,d0/d1)
AreaEllipse = -186 ; (RastPort,XPos,YPos,a,b)(a1,d0-d3)
AreaEnd     = -264 ; (RastPort),(a1)
AreaMove    = -252 ; (RastPort,XPos,YPos)(a1,d0/d1)

* Area Info            ; mit dieser Struktur arbeitet der
  struktur             ; Blitter bei allen Area-
Operationen
    aptr ai_VctrTbl  ; Sie wird mit InitArea
initialisiert
    aptr ai_VctrPtr  ; Wichtig ist es, die
VektorTabelle
    aptr ai_FlagTbl  ; ausreichend zu dimensionieren
    aptr ai_FlagPtr  ; Auf die Tabelle verweist
ai_VctrTbl
    word ai_Count
    word ai_MaxCount
    word ai_FirstX
    word ai_FirstY
  struktLaenge ai_size


AreaInfoAnlegen    ; Param: d0: Anzahl Koordinatenpaare
*--------------      Ergebnis in d0 (<>0 erfolgreich)
  movem.l d2/d3/a2/a3,-(sp)
  move.l d0,d3      ; Anz Koordinatenpaare aufbewahren
  AllocMemory #ai_size,#Mem_Clear ; AreaInfo-Struktur
  tst.l d0                        ; Struktur angelegt ?
  beq.s Fehler                    ; nein

  move.l RastPort(globals),a3 ; Zeiger auf RastPort
  move.l d0,rp_areaInfo(a3)   ; Struktur eintragen
  move.l d0,a2                ; Zeiger auf AreaInfo

  move.l d3,d2                ; Anz Koordinatenpaare
  mulu   #5,d2                ; 5 Bytes pro Paar
  AllocMemory d2,#Mem_Clear   ; Vektortabelle anlegen
  tst.l d0                    ; Tabelle angelegt
  bne.s Weiter                ; ja

  FreeMemory a2,#ai_size      ; AreaInfo aufgeben
  clr.l rp_areaInfo(a3)       ; in RastPort vermerken
  clr.l d0                    ; Fehler signalisieren
  bra.s Fehler
Weiter
  move.l d0,a1                ; Adr Vektortabelle
  move.l a2,a0                ; Adr AreaInfo
  move.l d3,d0                ; Anz Koordinatenpaare
  CallGraph InitArea
Fehler
  movem.l (sp)+,d2/d3/a2/a3
  rts



AreaInfoAufgeben
*---------------
  move.l a2,-(sp)
  move.l RastPort(globals),a2   ; RastPort holen
  move.l rp_areaInfo(a2),a2     ; Zeiger auf AreaInfo

  move.l #5,d1                  ; 5 Bytes/Vektor

  mulu   ai_MaxCount(a2),d1     ; Vektortab. in Bytes

  FreeMemory ai_VctrTbl(a2),d1  ; Tabelle aufgeben

  FreeMemory a2,#ai_size        ; AreaInfo aufgeben
  clr.l rp_areaInfo(a2)        ; in RastPort vermerken
  move.l (sp)+,a2
  rts


PolygonUndEllipse
*----------------
  movem.l a2/d2-d3,-(sp)
  move.l  RastPort(globals),a2 ; RastPort holen

  move.l  a2,a1                ; 1. Koordinate Polygon
  move.l  #360,d0              ; XPos
  move.l  #120,d1              ; YPos
  CallGraph AreaMove           ; Ausgangspos. setzen

  move.l  a2,a1                ; 1. Vektor ermitteln
  move.l  #420,d0
  move.l  #150,d1
  CallGraph AreaDraw

```

Listings zum Assemblerkurs

```
         move.l  a2,a1                ; 2. Vektor
         move.l  #380,d0
         move.l  #200,d1
         CallGraph AreaDraw

         move.l  a2,a1                ; 3. Vektor
         move.l  #340,d0              ; Ausgangspunkt gleich
         move.l  #140,d1              ; Endpunkt wird selbst
         CallGraph AreaDraw           ; erkannt

         move.l  a2,a1                ; RastPort
         move.l  #100,d0              ; Mittelpunkt
         move.l  #140,d1
         move.l  #70,d2               ; Radien
         move.l  #70,d3
         CallGraph AreaEllipse

         FaerbeAPen #14          ; Farbe fuer beide Figuren
         move.l a2,a1
         CallGraph AreaEnd       ; beide Figuren zeichnen
         movem.l (sp)+,a2/d2-d3
         rts
```

Listing 5: Fuer die Arbeit mit den Area-Routinen bedarf es einer initialisierten AreaInfo-Struktur und einer Vektortabelle. Der Aufwand wird durch schnellste Grafik belohnt. Eine Kostprobe bietet "PolygonUndEllipse".

```
SetAfPt macro ; RastPort mit Fuellpattern versehen
*------------     Params: /1 Adr RasPort, /2 Adr Muster
               ; /3 Musterhoehhe (Exponent Zweierpotenz)
   move.l \1,a1             ; RastPort holen
   move.l \2,rp_areaPtrn(a1) ; Muster eintragen
   move.b \3,rp_areaPtSz(a1) ; Musterhoehe eintragen
   endm

...                        ; so ist SetAfPt einzusetzen
   move.l RastPort(globals),a1
   lea.l  Muster0(pc),a0
   SetAfPt a1,a0,#3        ; Muster bei RastPort anmelden

   bsr PolygonUndEllipse      ; Muster anwenden

   SetAfPt #0,#0,#0           ; Muster zuruecksetzen
...

Muster0 dc.w %1100110011001100 ; als Fuellmuster
        dc.w %0110011001100110 ; es sind 2^3 Woerter
```

Listings zum Assemblerkurs

```
              dc.w %0011001100110011 ; somit ist 3 der
              dc.w $FF               ; Parameter fuer die
              dc.w $FF               ; Hoehe des Musters
              dc.w %0011001100110011
              dc.w %0110011001100110
              dc.w %1100110011001100
```

Listing 6: "SetAfPt" fuettert die RastPort-Struktur mit den Informationen ueber ein Pattern, das bei Fuelloperationen zu verwenden ist.

```
SetzePunkt ; Parameter: a0 BitMap-Adr
*---------               d0 XPos
                       ; d1 YPos
                       ; d2 Farbe
   lea.l  bm_planes(a0),a0 ; erste BitPlane
   move.w d0,d3            ; XPos merken
   lsr.w  #3,d0            ; XPos/8 = Byte der Zeile

   mulu   #80,d1           ; Zeilenaddr. in BitPlane
   add.w  d0,d1            ; Adr Byte in dem Punkt liegt
   and.w  #7,d3            ; Pos. des Punktes im Byte

   subq.w #7,d3            ; BitNr ermitteln
   neg.w  d3               ; BitNr ist stets positiv

   and.w  #$F,d2           ; auf 16 Farben begrenzen
                           ; entspr. 4 BitPlanes
\LO
   btst   #0,d2            ; Punkt in aktueller Plane
                           ; zu setzen ?
   beq.s  \L1              ; nein

   move.l (a0),a1          ; aktuelle Plane holen
   adda.l d1,a1            ; ergibt absolute Zeilenaddr.
   bset   d3,(a1)          ; Punkt setzen

\L1
   addq.l #4,a0          ; auf naechste BitPlane
   lsr.b  #1,d2          ; naechste Farbebene nach Bit 0
   tst.b  d2             ; noch Farbebenen zu bearbeit. ?
   bne.s  \LO            ; ja
   rts

SetzePunktDemo
*-------------
   movem.l a2/d2-d7,-(sp)
   move.l #600,d4          ; StartPosition X
   move.l #200,d5          ; Y
```

Listings zum Assemblerkurs

```
        move.l #150,d6             ; Farben
        move.l d6,d7               ; Anzahl zu setzende Punkte
        move.l RastPort(globals),a2
\L
        move.l rp_bitMap(a2),a0     ; Adr BitMap
        move.w d4,d0                ; XPos
        move.w d5,d1                ; YPos
        move.w d6,d2                ; Farbe
        bsr SetzePunkt
        subq.w #1,d4                ; neue XPos
        subq.w #1,d5                ; neue YPos
        subq.w #1,d6                ; neue Farbe
        dbeq d7,\L
        movem.l (sp)+,a2/d2-d7
        rts
```

Listing 7: Superschnelle Punkte mit einer massgeschneiderten Routine ohne Overhead, aber auch ohne Schutz vor Ounbeabsichtigtem Ueberschreiben des Speichers.

```
* Die nachfolgenden Vereinbarungen gehoerten
* in eine Datei namens "incl/Hardware.i"

custom    = $0DFF000 ; Hardwareadresse
                     ; Custom-Chip
bltddat   = $000     ; Offsets fuer Register
dmaconr   = $002     ; des Blitters
bltcon0   = $040
bltcon1   = $042
bltafwm   = $044
bltalwm   = $046
bltcpt    = $048
bltbpt    = $04C
bltapt    = $050
bltdpt    = $054
bltsize   = $058
bltcmod   = $060
bltbmod   = $062
bltamod   = $064
bltdmod   = $066
bltcdat   = $070
bltbdat   = $072
bltadat   = $074
dmacon    = $096

LINEMODE = $001   ; versetzt Blitter in Linienmodus
SIGNFLAG = $040   ; bei negativen Steigungen benoetigt

BLITTER  = 06     ; Auskunft ueber Blitter-Status
BLTDONE  = 14

OCTANT1  = 16     ; Oktanten, blittergerecht bezeichnet
OCTANT2  = 00
OCTANT3  = 08
OCTANT4  = 20
OCTANT5  = 28
OCTANT6  = 12
OCTANT7  = 04
OCTANT8  = 24


ZieheLinie ; mit dem Blitter
*---------      Para: d0 = x1
                    ; d1 = y1
                    ; d2 = x2
                    ; d3 = y2
                    ; a3 = BitMap (mit 4 Planes)
                    ; d7 = Farbe
x1 equr d0
x2 equr d2
y1 equr d1
y2 equr d3

dx equr d2
dy equr d3

* Oktant ermitteln zur Unterstuetzung des Blitters

   lea.l X1_kl_X2(pc),a0
   sub.w x1,x2               ; x2 - x1 = dx
   bpl.s \y                  ; dx positiv ?
   addq.l #4,a0              ; auf naechste Oktantenzeile
   neg.w dx                  ; falls dx negativ
\y
   lea.l Y1_kl_Y2(pc),a1
   sub.w y1,y2               ; y2 - y1 = dy

   bpl.s \d
   addq.l #4,a1
   neg.w  dy
\d
   lea.l DX_gr_DY(pc),a2
   cmp.w dy,dx               ; dx > dy ?
   bpl.s \d1                 ; ja
   addq.l #4,a2
   exg dx,dy                 ; Algorithmus erwartet dx > dy
\d1
   move.l (a0),d5  ; Schnittmenge der 3 Oktantenmengen
   and.l  (a1),d5  ; bilden.
   and.l  (a2),d5  ; Ergebnis : Gesuchter Oktant
   tst.w  d5       ; Oktant im Langwort so verschieben,
   bne.s  \s       ; dass er im unteren Nibble landet
   swap   d5
\s
   tst.b  d5
   bne.s  \s1
   asr.w  #8,d5    ; Motorola Syntax : #0
\s1
   cmp.b  #$F,d5
   bmi.s  \s2
   asr.b  #4,d5
\s2                ; Oktant nun im unteren Nibble
   lea.l Oktanten(pc),a0 ; Oktant als Offset verdoppeln
   add.w d5,d5           ; da Tabelle wortweise angelegt
   move.w 0(a0,d5.w),d5  ; Oktant blittergerecht holen
                         ; einschliesslich des Linemodus
\Linie
   move.l #80,d4 ; spezial. Routine auf Breite 640
   mulu.w d4,y1  ; ergibt Zeilenaddr innerhalb Plane

* x1 zerlegen in Byte und Pixel

   move.l x1,d6
   lsr.w  #3,d6        ; Nr. des Bytes in der Zeile
   add.w  d6,y1        ; Addr. des Bytes bezogen auf Plane
   move.l y1,a2        ; fuer andere Planes merken
   and.w  #7,d0        ; Pixel innerhalb des Bytes

con0   equr d0

   ror.w  #4,con0      ; obere 4 Bits enthalten Pixelpos.
                       ; nun die DMA-Kanaele zuweisen
   or.w   #$BFA,con0   ; B40 zeigt Arbeitsart d. Blitt.,
                       ; Fenster fuer Fenster als Treppe
   lsl.w  #2,d3        ; 4 * dy
   add.w  d2,d2        ; 2 * dx

blt equr d1

   move.w d2,blt       ; Wort Blitterstart zusammensetzen
   lsl.w  #5,blt       ; 2 * dx ( = Linienlaenge setzen )
   add.w  #$42,blt     ; Bits 6 u. 1 setzen (Hoehe,
Breite)

   lea.l custom,a1       ; Blitter addressieren
   btst  #BLITTER,dmaconr(a1) ; empfohlener Testaufruf
\ErwarteBlitter
   btst   #BLITTER,dmaconr(a1)
   bne.s  \ErwarteBlitter

   move.w #$8400,dmaconr(a1)  ; %1000010000000000
                              ; ¦    ¦
                              ; ¦    ¦  = Blitter DMA Prio-

                              ; ¦       ritaet ueber CPU
                              ; = Prioritaet setzen
   move.w dy,bltbmod(a1)      ; 4 * dy
   sub.w  d2,d3               ; 4 * dy - 2 * dx
   ext.l  d3
   move.l d3,d6               ; Steigung fuer BLTAPT
   bpl.s  \w
   or.w   #SIGNFLAG,d5        ; zu setzen, wenn fallend
\w
   move.w d4,bltcmod(a1)   ; Modulo (Bildbreite), fuer
   move.w d4,bltdmod(a1)   ; gerade und ungerade Planes
   sub.w  d2,d3            ; 4 * dy - 4 * dx
   move.w d3,bltamod(a1)   ; Steigung
   move.l #$FFFFFFFF,bltafwm(a1) ; First Word Mask
Kanal A
                           ; fuer Linien alles 1
   move.w #$8000,bltadat(a1) ; so im Line-Modus
   move.w #$FFFF,bltbdat(a1) ; Maske auf 111...
   move.w #3,d2              ; 4 Bitplanes bearbeiten
\loop
   move.l (a3),a0           ; Addr Bitplane holen
   adda.l a2,a0             ; Byteaddr innerh. Plane dazu
\warte
   btst   #BLITTER,dmaconr(a1)
   bne.s  \warte


   move.b #$FA,con0         ; Annahme Punkt setzen
   btst   #0,d7             ; Punkt in Plane setzen ?
   bne.s  \n                ; ja
   move.b #$0A,con0         ; kein Punkt in dieser Plane
\n                          ; zu setzen aber loeschen
   move.l d6,bltapt(a1)
   move.l a0,bltcpt(a1)      ; Linie, Anfangsaddr.
   move.l a0,bltdpt(a1)      ; Linie, Anfangsaddr.
   move.w con0,bltcon0(a1)   ; Anfangspos + DMA-Zugriff
   move.w d5,bltcon1(a1)
   move.w blt,bltsize(a1)    ; Starte Blitter

   addq.l #4,a3           ; auf naechste Plane
   lsr.b  #1,d7           ; naechste Farbebene ins Bit 0
   dbra.s d2,\loop

\wart
   btst   #BLITTER,dmaconr(a1)
   bne.s  \wart

   move.w #$0400,dmaconr(a1) ; %0000010000000000
                             ; ¦    ¦
                             ; ¦    = Blitter DMA Prio-
                             ; ¦      ritaet ueber CPU
   rts                       ; = Prioritaet aufheben


AreaBlitterDemo
*--------------
...
   movem.l d2-d7/a2-a5,-(sp)
   move.l #20,d1              ; x1
   move.l #160,d1             ; y1
   move.l #600,d2             ; x2
   move.l #40,d3              ; y2
   move.w #6,d7               ; Farbe violett
   move.l RastPort(globals),a2
   move.l rp_bitMap(a2),a0    ; Adr BitMap
   lea.l  bm_planes(a0),a3    ; erste BitPlane

   bsr ZieheLinie
   movem.l (sp)+,d2-d7/a2-a5
   rts

   cnop 0,4

* Tabelle der Oktanten (1 - 8)
X1_kl_X2 dc.l $12000078
         dc.l $00345600
Y1_kl_Y2 dc.l $12340000
         dc.l $00005678
DX_gr_DY dc.l $10045008
         dc.l $02300670

* Tabelle zur blittergerechten Umwandlung
* der Oktanten bei gleichzeitiger Addition
* des LINEMODE

Oktanten
          dc.w 0                ; dummy
```

Listings zum Assemblerkurs

```
                dc.w OCTANT1+LINEMODE  ; Zusammenschreiben!
                dc.w OCTANT2+LINEMODE
                dc.w OCTANT3+LINEMODE
                dc.w OCTANT4+LINEMODE
                dc.w OCTANT5+LINEMODE
                dc.w OCTANT6+LINEMODE
                dc.w OCTANT7+LINEMODE
                dc.w OCTANT8+LINEMODE

Listing 8 : Linien mit dem Blitter erfordern einen hohen
            Programmieraufwand. Die Muehe lohnt sich, wenn
            es auf die Ausfuehrungsgeschwindigkeit ankommt.
```

```

* Oktant ermitteln zur Unterstuetzung des Blitters

   lea.l X1_kl_X2(pc),a0
   sub.w x1,x2                ; x2 - x1 = dx
   bpl.s \y                   ; dx positiv ?
   addq.l #1,a0               ; auf naechste Oktantenzeile
   neg.w dx                   ; falls dx negativ
\y
   lea.l Y1_kl_Y2(pc),a1
   sub.w y1,y2                ; y2 - y1 = dy
   bpl.s \d
   addq.l #1,a1
   neg.w  dy
\d
   lea.l DX_gr_DY(pc),a2
   cmp.w dy,dx                ; dx > dy ?
   bpl.s \d1                  ; ja
   addq.l #1,a2
   exg dx,dy               ; Algorithmus erwartet dx > dy
\d1
   move.b (a0),d5 ; Schnittmenge der 3 Oktantenmengen
   and.b  (a1),d5 ; bilden.
   and.b  (a2),d5 ; Ergebnis : Gesuchter Oktant

   move.w #8,d4
\l
   asr.b  #1,d5   ; schiebt ins carry
   dbcs.s d4,\l   ; nimmt Werte von 8 - 1 an
   move.w d4,d5   ; enthaelt nun Oktant
\s2
   lea.l Oktanten(pc),a0 ; Oktant als Offset verdoppeln
   add.w d5,d5              ; da Tabelle wortweise angelegt
   move.w 0(a0,d5.w),d5  ; Oktant blittergerecht holen
                            ; einschliesslich des Linemodus
\Linie
     .....

* Tabelle der Oktanten (1 - 8)

X1_kl_X2 dc.b %11000011 ; Okt1--> Okt8
         dc.b %00111100 ; entspr. $12345678
Y1_kl_Y2 dc.b %11110000
         dc.b %00001111
DX_gr_DY dc.b %10011001
         dc.b %01100110

Listing 9 : Die Oktanten nun als eine Menge von Bits.
            Der Codeteil kann entsprechende Teile in
            Listing 8 ersetzen, um an Ausfuehrungszeit zu
            gewinnen und das Programm zu kuerzen.
```

```

   sub.w x1,x2      ; ergibt delta x (dx)
   bmi.s \Okt3456   ; x2/y2 in Oktant 3-6
                    ; sonst 1,2,7 oder 8
   sub.w y1,y2      ; bilde dy
   bmi.s \Okt78     ; x2/y2 in 7 oder 8
   cmp.w dy,dx
   bmi.s \Okt2      ; dy > dx, folglich Oktant 2
   moveq.l #OCTANT1+LINEMODE,d5
   bra.s \Linie
\Okt2
   exg   dx,dy      ; Algorithmus erwartet dx groesser dy
   moveq.l #OCTANT2+LINEMODE,d5
   bra.s \Linie
\Okt78
   neg.w dy         ; Absolutwert dy bilden
   cmp.w dy,dx
   bmi.s \Okt7
   moveq.l #OCTANT8+LINEMODE,d5
   bra.s \Linie
\Okt7
   exg   dx,dy      ; Algorithmus erwartet dx groesser dy
   moveq.l #OCTANT7+LINEMODE,d5
   bra.s \Linie
\Okt3456
   neg.w dx         ; Absolutwert bilden
   sub.w y1,y2      ; ergibt dy
   bmi.s \Okt56     ; sonst Oktant 3 oder 4
   cmp.w dy,dx
   bmi.s \Okt3
   moveq.l #OCTANT4+LINEMODE,d5
   bra.s \Linie
\Okt3
   exg   dx,dy      ; Algorithmus erwartet dx groesser dy
   moveq.l #OCTANT3+LINEMODE,d5
   bra.s \Linie
\Okt56
   neg.w dy         ; Absolutwert dy bilden
   cmp.w dy,dx
   bmi.s \Okt6
   moveq.l #OCTANT5+LINEMODE,d5
   bra.s \Linie
\Okt6
   exg   dx,dy      ; Algorithmus erwartet dx groesser dy
   moveq.l #OCTANT6+LINEMODE,d5

     Listing 10:Alternativer Ansatz, den Oktanten fuer den
                Blitter zu bestimmen. Routine kann den Teil
                von "ZieheLinie" vor der Marke "\Linie" in
                Listing 8 ersetzen. Welche Loesung ist fuer
                Sie anschaulicher ?
                Welche ist performanter ?
```

Listings zum Assemblerkurs

Amiga-BASIC Programmieren mit Erfolg

Hallo, hallo – Amiga ruft BASIC

Im letzten Kursteil kommen noch einige Schmankerl auf Sie zu. Wir beschäftigen uns mit einem Grundprinzip modularer Programmierung, hören, wie unser Amiga mit uns spricht, und zaubern nebenbei noch Töne aus dem Lautsprecher heraus.

Kursfahrplan

Teil 1: BASIC zum Kennenlernen

Teil 2: Dem Spaghetti-Code keine Chance

Teil 3: Vom Strich zum Bild

Teil 4: Dateien über Dateien

Teil 5: Unter dem Programm: ein Unterprogramm

Ein Ziel sollte jeder vor Augen haben, der sich etwas intensiver mit Programmiersprachen, genauer gesagt mit der Entwicklung von Programmen, beschäftigt: Programme modular zu entwerfen, daß heißt, das Programm in möglichst kleine, eigenständige Programmteile aufzuspalten, die so jederzeit einzeln getestet werden können.

Hallo, Hauptprogramm ruft Unterprogramm

Ein weiterer Punkt ist, Programmteile, die ständig wieder gebraucht werden, zu einer Routine zusammenzufassen und als Unterprogramm laufen zu lassen. Der Vorteil solcher Routinen liegt klar auf der Hand: Sie sparen eine Menge an Zeit, Aufwand und auch Speicherplatz. Die Programme werden dadurch wesentlich übersichtlicher, besonders in bezug auf die Fehlerkontrolle. Wie sieht das Ganze nun programmiertechnisch aus?

Unterprogramme sind in sich geschlossene Programmteile oder auch logisch zusammengehörige Einheiten. Solche Einheiten lassen sich auf verschiedene Art und Weise definieren: Die kürzeste Form eines Unterprogramms ist die Funktion. In einer Funktion können beispielsweise komplizierte Formeln, deren Ergebnis als konstanter Wert im Programm eingebunden werden soll, definiert und berechnet werden. Jedoch ist zu beachten, daß die Funktionsdefinition nicht mehr als 80 Zeichen (entspricht einer Programmzeile in BASIC) umfassen darf. Die Syntax dieses Befehls lautet:

DEF FNName[(Arg[,Arg]...)] = Funktionsdefinition

Kommen wir zu den einzelnen Parametern: "Name" ist der Name der Variablen, der als Funktionsname dient und ohne Leerstelle hinter FN stehen muß. "Arg" ist der Variablenname, der beim Aufruf der Funktion durch die aktuelle Variable bzw. den aktuellen Wert ersetzt wird. Der Parameter "Funktionsdefinition" beinhaltet eine Rechenvorschrift oder Rechenformel, nach der der Funktionswert berechnet werden soll. Es kann jeweils nur **eine** Rechenvorschrift in einer Funktionsdefinition angegeben werden. Die Funktionsdefinition kann ein numerischer oder ein Zeichenkettenausdruck sein. Dabei ist jedoch darauf zu achten, daß dieser Ausdruck vom Typ her mit dem Typ von "Name" übereinstimmt.

Ein kurzes Beispiel soll die Arbeitsweise des Befehls DEF FN verdeutlichen. Begeben wir uns in die mathematischen Gefilde und erinnern uns an unseren Mathelehrer, der uns immer wieder mit der Frage nach dem Satz des Pythagoras gelöchert hat. Na, wissen Sie ihn noch? Richtig, im rechtwinkligen Dreieck ist das Quadrat über der Hypothenuse so groß wie die Summe der Quadrate über den Katheten. Demzufolge können wir nun eine Funktion entwerfen, die die Länge der Hypothenuse berechnen soll. Ausgehend von unserer Pythagoras-Formel ($a^2 + b^2 = c^2$) gehen wir in unseren BASIC-Interpreter und geben als Definiton folgendes ein:

```
DEF FNc(a,b)=SQR((a*a)+(b*b))
```

Möchte man nun verschiedene Eingaben machen, geben wir noch folgendes ein:

```
INPUT "Geben Sie Werte ein";a,b
PRINT "Die Länge der Hypothenu-
se beträgt:"FNc(a,b)
```

Damit haben wir ein kurzes Programmbeispiel erstellt, mit dem sich bereits die verschiedenen Längen unserer Hypothenusen berechnen lassen. Ein weiteres kleines Beispiel zeigt, wie in einer Endlosschleife viele unterschiedliche Rechtecke per Rundungsfaktor unter Verwendung der obengenannten Funktion (DEF FNc...) erzeugt werden. Gehen Sie wieder in das LIST-Fenster Ihres BASIC-Interpreters, und geben Sie folgendes Beispiel ein:

```
start:
DEF FNc=SQR((a*a)+(b*b))
a=INT(RND*100)+10
b=INT(RND*100)+10
AREA (0,0)
AREA (INT(FNc),0)
AREA (INT(FNc),INT(FNc))
AREA (0,INT(FNc))
AREAFILL x
IF x=0 THEN x=1 ELSE x=0
GOTO start
```

Da dies wie gesagt eine Endlosschleife ist, können Sie dieses Beispiel in der Menüleiste Ihres BASIC-Interpreters mit STOP im RUN-Menü beenden. Falls Sie jedoch ein wenig herumprobieren möchten, ändern Sie dieses Miniprogrämmchen doch einfach so ab, daß Sie beispielsweise ein Ende-Kriterium hinzufügen oder ähnliches. Da diese Funktionen jedoch auf eine Programmzeile beschränkt sind, wenden wir uns einer geläufigeren und komfortableren Möglichkeit, Unterprogramme zu erstellen, zu: der GOSUB-label-Konstruktion.

Sprünge hin und her im Programm

Mit diesem Konstrukt kann von einem Hauptprogramm aus in Subroutinen verzweigt und auch wieder zurück ins Hauptprogramm gesprungen werden, wenn der Interpreter auf ein RETURN trifft, das die Subroutine beendet. Die Anzahl der Subroutinen, die in einem Programm verwendet werden, ist unbegrenzt, jedoch kennen diese Routinen keine lokalen Variablen. Das heißt, alle Variablen, die in einer Subroutine enthalten sind, sind auch im Hauptprogramm bekannt. Hier liegt aber auch "der Hase im Pfeffer": Bei der Definition der einzelnen Variablen muß man unbedingt dafür sorgen, daß diese Variablen nicht bereits im Hauptprogramm verwendet wurden, da dies sehr fatale Folgen hätte. Wird eine Variable, die im Hauptprogramm definiert ist, auch für die Subroutine neu definiert, ändert sie somit ihren Wert und demzufolge auch den Wert, den sie im Hauptprogramm hat.

Damit ist gleichzeitig die Funktionsweise solcher Subroutinen stark eingeschränkt, da sie nicht als eigenständiges Programm lauffähig sind, sondern immer nur im Zusammenhang und in Verbindung mit dem Hauptprogramm laufen können. Auch die Fehlersuche ist ein kritischer Faktor, den man bei der Verwendung von Subroutinen in eigenen Programmen bedenken sollte. Innerhalb von Subroutinen können jeder Zeit noch weitere Subroutinen aufgerufen werden. Hier setzt lediglich der Hauptspeicherplatz diverse Grenzen. Nach der RETURN-Anweisung kehrt die Subroutine wieder zum Hauptprogramm zurück und setzt direkt mit der nächsten Anweisung, die der GOSUB label-Anweisung folgt, fort. Die An-

wendung solcher Anweisungen ist mit einer gewissen Vorsicht zu genießen, da die übrigen GOSUB-, CALL-, WHILE- und FOR-Anweisungen, die zur Zeit des Aufrufs der Subroutine aktiv waren, auch hinterher aktiv bleiben. Das heißt, sie belegen somit Bezugsadressen auf dem Stapelspeicher.

Die Syntax der Anweisung GOSUB label sieht folgendermaßen aus:

```
GOSUB label
...
...
RETURN
```

Listing 1 zeigt in einem einfachen Beispiel die Funktionsweise dieser Anweisung.

Hallo, hallo, rufe Unterprogramm

Wenden wir uns nun den "richtigen" Unterprogrammen zu. Richtig insofern, da diese Unterprogramme auch tatsächlich in sich geschlossene Einheiten bilden. Es können in der Tat lokale (im wahrsten Sinne des Wortes) Variablen verwendet werden, die **nur** dem Unterprogramm, jedoch nicht dem Hauptprogramm bekannt sind. Dies führt natürlich auch dazu, daß Variablen, die im Hauptprogramm verwendet werden, auch im Unterprogramm verwendet werden können, ohne sich gegenseitig "ins Gehege zu kommen". Diese Unterprogramme können, da sie ja eigenständigen Charakter besitzen, alleine stehen und somit auch einzeln getestet werden. Die Fehlersuche läßt sich so gut eingrenzen. Ein weiterer Vorteil der sogenannten Unterprogrammtechnik liegt darin begründet, daß hier wirklich modulares Programmieren möglich ist. Das heißt, man kann sich selbst seine eigene Bibliothek zusammenstellen, indem man häufig wiederkehrende Routinen zu Unterprogrammen zusammenfaßt.

Schauen wir uns zunächst die Syntax dieser Unterprogramm-Technik an. Wir sprechen im folgenden nicht mehr vom Unterprogramm, sondern verwenden den passenderen Ausdruck SUB-Programm. Der Aufruf eines SUB-Programms erfolgt mit

```
CALL Name (Parameterliste)
```

wobei "Name" der Name des aufzurufenden SUB-Programms ist und "Parameterliste" die zu übergebenden Variablen bezeichnet. Die Syntax der eigentlichen Unterprogrammanweisung lautet dann:

```
SUB Name (Parameterliste) STATIC
END SUB
EXIT SUB
```

Gehen wir näher auf die einzelnen Parameter ein: Mit "Name" ist die Bezeichnung des jeweiligen SUB-Programm-Aufrufs gemeint. "Name" kann bis zu 40 Zeichen umfassen, darf jedoch in keiner anderen SUB-Anweisung verwendet werden. In "Parameterliste" können einfache oder Feldvariablen (Stichwort: Dimensionierung) angegeben werden, die Anzahl ist lediglich durch die Länge einer logischen Programmzeile (254 Zeichen) begrenzt. Die einzelnen Parameter werden durch Kommata getrennt. Bei Feldvariablen ist darauf zu achten, daß diese in runde Klammern "()" gesetzt werden. Ein wesentlicher Aspekt kommt dem Begriff "STATIC" zu. ("STATIC" heißt statisch oder unverändert.) Dieser besagt, daß zwischen zwei SUB-Programmaufrufen alle Variablen ihre Werte behalten. Ein weiterer wichtiger Punkt ist, daß alle Variablen vom Hauptprogramm **nicht** verändert werden können. Zudem setzt "STATIC" fest, daß das SUB-Programm nicht rekursiv ist, das heißt, sich selbst und andere SUB-Programme aufrufen kann. "END SUB" bedeutet, daß das SUB-Programm beendet ist und das die Programmsteuerung an das Hauptprogramm zurückgegeben wird. Dabei wird die dem SUB-Programmaufruf folgende Anweisung im Hauptprogramm ausgeführt. Mit der "EXIT SUB"-Anweisung ist es möglich, aus dem SUB-Programm "auszusteigen". Diese Art Aussprung ist beispielsweise bei bedingten Abfragen angeraten. Wir wollen aber nicht näher darauf eingehen.

Ein kleines Beispiel soll zeigen, ob Variablen in einer "SUB STATIC"-Anweisung verändert werden oder nicht. Gehen Sie in Ihren BASIC-Interpreter, und geben Sie im LIST-Fenster ein:

```
a=1
b=2
c=3
CALL Unter
PRINT a,b,c
FOR i=1 TO 5000:NEXT
END
SUB Unter STATIC
a=5:b=6:c=7
END SUB
```

Sie werden am Ergebnis feststellen, ob und wo sich etwas ändert. Zum Gegensatz tippen Sie bitte folgendes in Ihren BASIC-Interpreter ein:

```
a=1
b=2
c=3
CALL Unter
PRINT a,b,c
FOR i=1 TO 5000:NEXT
END
SUB Unter STATIC
SHARED c
a=5:b=6:c=7
END SUB
```

Sie werden sicherlich bemerkt haben, daß hier ein deutlicher Unterschied besteht. Das erste Beispiel zeigt, daß Variablen, die im Hauptprogramm definiert wurden, nicht durch das SUB-Programm verändert werden, da diese Variablen **lokale** Variablen sind und somit nur für ihren speziellen Bereich gelten, wo Sie definiert wurden. Demgegenüber zeigt das zweite Beispiel, wie mit Hilfe der "SHARED"-Anweisung eine Variable einen **globalen** Charakter erhält. Das heißt, diese Variable ist gleichzeitig im Hauptprogramm und im SUB-Programm gültig.

Ein wichtiger Hinweis: Folgende Anweisungen dürfen in SUB-Programmen auf gar keinen Fall verwendet werden:

- *CLEAR*
- *COMMON*
- *DECLARE FUNCTION*
- *DEF FN*
- *SUB STATIC*
- *LIBRARY (sollte aus Zeitgründen nicht in einem SUB-Programm verwendet werden).*

Damit wollen wir es beim Thema "Unterpogamme" bewenden lassen und uns einigen akustischen Bonbons zuwenden.

Sie haben sich Ihren Amiga bestimmt nicht nur gekauft, um damit damit zu spielen. Sicherlich waren Sie genauso neugierig wie ich, was man mit der "Spielekiste" noch so alles anstellen kann. Eine große Stärke haben wir ja bereits kennengelernt: die grafischen Fähigkeiten des Amiga. Damit nicht genug, nein, unser Amiga kann sich richtiggehend mit uns unterhalten. Und, welcher "stinknormale" PC kann das schon?

Hilfe, mein Computer spricht mit mir!

Praktisch ist die ganze Sache auch noch, denn in AmigaBASIC ist die Sprachausgabe fast ein Kinderspiel. Mit dem "SAY"-Befehl kann Sprache in fast jeder beliebigen Landessprache ausgedrückt werden, vorausgesetzt man ist der phonetischen Lautschrift des entsprechenden Landes kundig. Die Syntax dieses Befehls lautet:

```
SAY Zeichenkette [,Modus]
```

Sie brauchen aber keine Angst zu haben, so schlimm, wie es gerade klingt, ist es nicht. Um etwas mit "SAY" sagen zu können, nimmt man sich zunächst eine "Zeichenkette" vor, die aus Phonem-Codes (Phonem entspricht dem deutschen "Laut"; es handelt sich also um Laut-Codes) zusammengesetzt ist. Diese ergeben dann den Text, den man hören möchte. Ist ja alles schön und gut, doch wo bekomme ich diese Phonem-Codes denn her, werden Sie fragen.

Diese Frage hat sicherlich ihre Berechtigung. Es gibt hier mehrere Möglichkeiten, die man in Betracht ziehen kann. Zum einen schlägt man in seinem Amiga-Basic-Handbuch nach. Dort ist der Sprachausgabe ein ganzes Kapitel gewidmet. Eine weitere Möglichkeit ist die Verwendung eines Wörterbuchs. Dabei ist es egal, ob man ein deutsches oder ein englisches Wörterbuch zu Rate zieht. Wesentlich ist nur, daß hinter jedem Wort in Klammern der phonetische Ausdruck steht. Diesen kann man beruhigt hinter dem SAY-Befehl eingeben. Die letzte Möglichkeit besteht darin, die Worte, die man per SAY-Befehl ausgeben will, so zu schreiben, wie man sie im wahrsten Sinne des Wortes auch spricht.

Kommen wir nun zum Parameter "Modus": Dieser Parameter entspricht einem Ganzzahlenfeld mit mindestens neun Elementen (es wird dabei von 0 bis einschließlich 8 gezählt). Beschäftigen wir uns kurz mit den einzelnen Elementen.

Element 0: Grundfrequenz in Hertz, dabei können Werte zwischen 65 und 320 angegeben werden.

Element 1: Betrifft die Modulation, hier können zwei Werte angegeben werden. 0 für Silbenmodulation und Betonung und 1 für eine monotone, roboterhafte Sprache.

Element 2: Regelt die Sprechgeschwindigkeit, wobei Werte zwischen 40 und 400 angegeben werden können.

BASIC-Glossar (Teil 4)

In unserem BASIC-Glossar finden Sie außer den Befehlen, die im Text beschrieben werden, auch artverwandte Befehle:

[CALL]Sprungmarke[(Argumentliste)]
oder
[CALL]num Var [(Argumentliste)]
Mit dieser Anweisung wird ein AmigaBASIC-Unterprogramm (mit SUB... definiert, siehe obere Syntax), ein Maschinenprogramm (siehe untere Syntax) oder eine Maschinensprache-Subroutine aus einer Bibliothek (mit der LIBRARY-Anweisung definiert, siehe untere Syntax) aufgerufen. Das Schlüsselwort CALL muß nicht unbedingt verwendet werden. Einige Beispiele:

- Aufruf von AmigaBASIC-Unterprogramm:

SUB Unterprogramm(a,b) STATIC
END SUB
CALL Unterprogramm(a,b)
Hier können Variablen oder Ausdrücke übergeben werden, wobei Variablen durch Referenz und Ausdrücke durch ihren Wert übergeben werden.

- Aufruf von Maschinenprogramm:

CALL Routine(VARPTR(x))
Dies ist die einzige Möglichkeit, die Programmsteuerung an ein externes Unterprogramm zu übertragen. Wichtig ist, daß die aufgerufene Maschinenroutine auf einer geraden Speicherstelle beginnt, da sich der Rechner sonst ins Nirwana verabschiedet.

- Aufruf von Bibliotheksroutinen:

LIBRARY"graphics.library"
CALL Draw(30,40)
Diese Bibliotheksroutinen sind spezielle Amiga-Programmdateien, die dynamisch, das heißt, während der Laufzeit, in AmigaBASIC eingebunden werden.

CLEAR[,[Prg][,Stap]]
Dieser Befehl dient dazu, die Inhalte aller Variablen entweder durch Setzen auf 0 oder durch Setzen eines Leerstrings zu löschen.

COMMON Variable[,Variable]...
Die bei dieser Anweisung angegebenen Variablen werden an ein Programm übergeben, das mit der CHAIN-Anweisung aufgerufen wurde. Werden Feldvariablen übergeben, so werden diese durch Klammern () hinter dem jeweiligen Namen ergänzt.

DECLARE FUNCTION Name[(Parameterliste)]LIBRARY
Mit dieser Anweisung wird eine Maschinensprache-Routine aus einer Bibliothek, die mit LIBRARY angegeben wird, als eine aufrufbare Funktion deklariert.

DEF FNName[(Arg[,Arg]...) = Funktionsdefiniton
Der Anwender kann eine eigenständige Funktion definieren und benennen.

END SUB
Dies bezeichnet das Ende eines Unterprogramms. Die Programmsteuerung wird an die Anweisung übergeben, die dem Unterprogrammaufruf im Hauptprogramm folgt.

EXIT SUB
Mit dieser Anweisung kann an jeder beliebigen Stelle das Unterprogramm verlassen werden. Auch hier geht die Programmsteuerung an die dem Unterprogrammaufruf folgende Anweisung im Hauptprogramm weiter.

v = FRE(x)
Diese Funktion liefert je nach dem Wert von x Aussagen über freien Speicherplatz in Bytes: Wenn x = -1 liefert die Aussage über den noch verfügbaren Speicherplatz
Wenn x = -2 liefert die Aussage über den noch verfügbaren AmigaBASIC-Stapelspeicher
Wenn x>-1 liefert die Aussage über den noch verfügbaren Speicher im AmigaBASIC-Datensegmebnt

GOSUB Label
..
RETURN [Weiter]
Mit dieser Anweisung erfolgt ein Sprung in eine Subroutine und die Rückkehr in das Hauptprogramm. Label ist der Name der betreffenden Subroutine. Weiter ist optional.

LIBRARY"Dateiname"
LIBRARY CLOSE
Mit dieser Anweisung wird eine Bibliothek mit Maschinensprache-Unterprogrammen für den Zugriff aus AmigaBASIC heraus geöffnet und mit LIBRARY CLOSE alle geöffneten Bibliotheken geschlossen.

SAY Zeichenkette[,Modus]
Gibt eine Zeichenkette von Laut-Codes in einer relativ verständlichen Sprache wieder, wobei mit dem Parameter "Modus" 9 verschiedene Elemente angegeben werden können, die die Ausgabe steuern.

SHARED Variable[,Variable]...
Mit dieser Anweisung können Variablen innerhalb eines mit SUB aufgerufenen Unterprogramms als global, das heißt, allgemein gültig, definiert werden.

SOUND Frequenz,Dauer[,[Laut][,Kanal]]
Mit dieser Anweisung wird ein Ton in einer Warteschlange abgelegt, die anschließend über einen entsprechend wählbaren Ton-Kanal abgespielt wird. Dabei sind jeweils Frequenz, Dauer und Laustärke dieses Tons frei wählbar.

SOUND WAIT
Diese Anweisung legt die Ergebnisse aller folgenden SOUND-Anweisungen in eine Warteschlange.

SOUND RESUME
Diese Anweisung beendet den Wartezustand der SOUND WAIT-Anweisung.

SUB Name[(Parameterliste)] STATIC
Dies Anweisung leitet ein AmigaBASIC-Unterprogramm ein. STATIC legt dabei fest, daß die im Unterprogramm verwendeten Variablen lokal sind.

v$ = TRANSLATE$(text)
Mit dieser Funktion werden Folgen von Laut-Codes erzeugt und an die SAY-Anweisung übergeben. Text ist dabei eine Zeichenkette aus der angloamerikanischen Sprache.

WAVE Kanal,Definition
Mit dieser Anweisung lassen sich für einen anzugebenden Tonkanal (Kanal) Formen von Tonwellen definieren.

Element 3: Hier kann das Geschlecht des Sprechers eingestellt werden. 0 entspricht einer männlichen und 1 einer weiblichen Stimme.

Element 4: Die Stimmlage kann verändert werden von 5000 (tief) bis 28000 (hoch).

Element 5: Damit läßt sich die Lautstärke regeln – von 0 (kein Ton) bis 65 (größte Lautstärke).

Element 6: Hier wird der Tonkanal für die Sprachausgabe eingestellt.

Element 7: Einstellung des Synchronisationsmodus – 0 entspricht synchroner, 1 asynchroner Sprachausgabe

Element 8: Dient zur Steuerung der Bearbeitung aufeinanderfolgender SAY-Anweisungen bei asynchroner Sprachausgabe. Es sind drei Werte möglich: 0 entspricht der normalen Verarbeitung, bei 1 wird die Sprachausgabe abgebrochen, und bei 2 wird die Bearbeitung der aktuellen SAY-Anweisung abgebrochen.

Damit die SAY-Anweisung nicht so ganz alleine im Raum steht, haben sich die Programmierer von Amiga-BASIC eine spezielle Funktion geschaffen: TRANSLATE.

Diese Funktion liefert einen String, der die dem Text entsprechenden Phonem-Codes enthält. Genauer gesagt, TRANSLATE spricht den Text aus, den man SAY übergibt. Die Syntax dieses Befehls sieht wie folgt aus:

v$ = TRANSLATE$(Text)

Sehen wir uns die Syntax etwas näher an. "v$" ist die Variable, die der SAY-Anweisung übergeben wird. In dem Parameter "Text" ist ein Zeichenkettenausdruck, dessen Inhalt in der bisher vorliegenden Form in angloamerikanischer Sprache angegeben werden muß und der anschließend mit der SAY-Anweisung ausgegeben wird.

Ein kleines Beispiel:

```
a$=TRANSLATE$("fishers frits
fisht frisha fisha")
SAY a$
b$=TRANSLATE$("frisha fisha
fisht fishers frits")
SAY b$
```

Das Ergebnis dürfte Ihnen sicherlich irgendwo schon einmal begegnet sein, zumindest verdeutlicht es Ihnen die Wirkungsweise der TRANSLATE-Funktion. Ein kleines Programm (Listing 2) zeigt Ihnen eine Möglichkeit, mit der Sprache in Amiga-BASIC realisiert werden kann. Mit einiger Übung läßt sich Ihr Amiga bestimmt zu einem wahren Sprachkünstler entwickeln.

Amiga – die Soundmaschine

Musik auf dem Amiga, wem klingen da nicht die Ohren. Mit Amiga-BASIC ist Musik sogar mit einem einzigen Befehl möglich: SOUND heißt das Zauberwort, mit dem sich sogar kleine Lieder spielen lassen. Wie funktioniert dieser SOUND-Befehl? Es wird ein Ton mit frei wählbarer Frequenz, Dauer und Lautstärke in einer sogenannten Ton-Warteschlange abgelegt, die anschließend über einen gleichfalls wählbaren Ton-Kanal abgespielt wird. Schauen wir uns die Syntax an:

SOUND Frequenz,Dauer[,[Laut][,Kanal]]
SOUND WAIT
SOUND RESUME

Die einzelnen Parameter haben folgende Bedeutung: Bei "Frequenz" kann ein Ganzzahlwert angegeben werden, der im Bereich zwischen 20 und 15000 liegen muß. Diese Werte entsprechen den Angaben in Hertz. Ein Beispiel: Ein Wert von 100 erzeugt unter Amiga-BASIC einen Ton von 100 Hertz. Die Tondauer wird mit dem Parameter "Dauer" geregelt, wobei hier ein numerischer Ausdruck im Bereich zwischen 0 und 77 angegeben werden muß. Mit "Laut" kann die Lautstärke des angegebenen Tons geregelt werden. Der Wert kann im Bereich von 0 (kleinste Lautstärke) bis 255 (größte Lautstärke) liegen. Mit "Kanal" können die Tonkanäle bestimmt werden, hier sind Werte zwischen 0 und 3 möglich.

Kommen wir zum Begriff "SOUND WAIT": Mit dieser Anweisung werden die Ergebnisse der nachfolgenden SOUND-Anweisungen in einer Warteschlange abgelegt, und zwar so lange, bis eine SOUND-RESUME-Anweisung erfolgt. Damit kann eine Synchronisation der Töne aus den unterschiedlichen Tonkanälen des Amiga erfolgen.

Um unseren Amiga-BASIC-Kurs noch ein wenig abzurunden, stellen wir Ihnen zum Abschluß ein Programm vor, mit dem sich Adressen verwalten lassen. Sie können den Quelltext dem Listing 3 entnehmen. Bei diesem Programm sind viele der im gesamten Kurs erlernten und angewandten Befehle und Anweisungen enthalten.

Damit ist unser Kurs "Amiga-BASIC – Programmieren mit Erfolg" an seinem Ende angelangt. Keine Angst!!! Das Kapitel Amiga-BASIC ist damit längst noch nicht abgeschlossen. Wir greifen uns in den nächsten Ausgaben immer wieder kleinere Schmankerl heraus und zeigen Ihnen, wie Sie damit umgehen können. Amiga-BASIC läßt Sie also auch in Zukunft grüßen. Bis demnächst und gut "Üb".

(vb)

Listings

```
000:718 REM ***********************************************
001:412 REM *                unterprogramm.bas            *
002:762 REM *  Dies ist ein Beispiel fuer den Aufruf von   *
003:748 REM *  von Unterprogrammen mit Hilfe der GOSUB-    *
004:474 REM *  Anweisung.                                  *
005:612 REM *  Sprache: AmigaBASIC                         *
006:718 REM ***********************************************
007:000
008:000
009:008 PRINT "Hallo Hauptprogramm ruft Unterprogramm"
010:952 FOR i=1 TO 10000:NEXT
011:633 GOSUB unter1
012:393 LOCATE 20,1
013:228 PRINT "Das Hauptprogramm meldet sich wieder zurueck"
014:952 FOR i=1 TO 10000:NEXT
015:503 CLS
016:822 LOCATE 10,30
017:932 PRINT "Tschuess"
018:500 END
019:316 unter1:
020:503   CLS
021:436   LOCATE 8,20
022:216   PRINT "Hallo, hier ist das erste Unterprogramm"
023:820   LOCATE 10,20
024:138   PRINT "          weiter mit Taste          "
025:444   WHILE INKEY$="":WEND
026:634   GOSUB unter2
027:982 RETURN
028:000
029:318 unter2:
030:503   CLS
031:836   LOCATE 12,20
032:408   PRINT "Juhu, hier meldet sich Unterprogramm zwei"
033:852   LOCATE 14,20
034:138   PRINT "          weiter mit Taste          "
035:444   WHILE INKEY$="":WEND
036:982 RETURN
037:026
```

Listing 1: Die Unterprogrammtechnik dargestellt am Beispiel einer GOSUB-Anweisung

```
000:718 REM ***********************************************
001:940 REM *                  sprache.bas                *
002:584 REM * Dieses Programm verdeutlicht die Sprachaus-  *
003:840 REM * gabe mit Hilfe der SAY-Anweisung             *
004:612 REM * Sprache: AmigaBASIC                          *
005:718 REM ***********************************************
006:000
007:727    WINDOW 1:CLS
008:074 start:
009:168    CLEAR
010:188    PRINT "Was soll ich sagen? Gib einen Satz ein!"
011:468    PRINT
012:112    INPUT Satz$
013:686    FOR n=0 TO 8
014:829       READ Feld%(n)
015:160    NEXT
016:273    Text$=TRANSLATE$(Satz$)
017:468    PRINT
018:949    PRINT Text$,Feld%
019:933    SAY Text$,Feld%
020:124    PRINT"Nochmal ?? (j/n)"
021:076    WHILE w$="":w$=INKEY$:WEND
022:380    IF UCASE$(w$)="J" THEN GOTO start
023:000
024:660 Feldwerte:
025:348    DATA 140,0,220,1,28000,64,11,1,0
026:026
```

Listing 2: So kann beispielsweise die Sprachausgabe in Amiga-BASIC aussehen

```
000:518 REM ***********************************************
        ********
001:892 REM *                    adressen.bas
        *
002:832 REM *               Adressverwaltungsprogramm
        *
003:396 REM *          (c) 1990 Steffen Kaiser & AMIGADOS
        *
004:612 REM *                  Sprache: Amigabasic
        *
005:518 REM ***********************************************
        ********
006:000
007:000
008:000
009:345 DIM nr(200)
010:000
011:201 GOSUB oeffne
012:164 GOSUB Fenster
013:920 GOSUB Farben
014:815 GOSUB Gadgets
015:055 GOSUB Maske
016:656 Marke:
017:983 GOSUB choose
018:961 ON ch GOSUB rueck,vor,change,watch,neu,Talk,Dru,loes
        ch,raus
019:491 GOTO Marke
020:500 END
021:000
022:000
023:452 oeffne:
024:800 OPEN "R",#1,"df0:telefon.dat",104
025:232 FIELD #1,20 AS n$,20 AS v$,20 AS s$,4 AS p$,20 AS o$
        ,20 AS t$
026:408 Anzahl = LOF(1)/104
027:982 RETURN
028:000
029:000
030:370 Fenster:
031:302 SCREEN 1,640,256,3,2
032:905 WINDOW 1,"Adressenverwaltung",(3,0)-(626,222),20,1
033:549 COLOR 5
```

```
034:971 PAINT (1,1)
035:316 COLOR 1,0
036:982 RETURN
037:000
038:890 Farben:
039:176 PALETTE 0,0,0,0
040:261 PALETTE 1,1,1,1
041:321 PALETTE 2,1,0,1
042:372 PALETTE 3,0,1,0
043:433 PALETTE 4,0,0,1
044:297 PALETTE 5,.1,.5,.5
045:565 PALETTE 6,0,1,1
046:759 PALETTE 7,.5,1,.3
047:982 RETURN
048:000
049:000
050:672 Gadgets:
051:039 '
052:526 '     Hier wird der Balken mit den Gadgets gezeichnet
053:039 '
054:194 LINE (1,154)-(629,155),6,bf
055:394 LINE (1,180)-(629,181),6,bf
056:526 FOR i=0 TO 8
057:507  x1=i*77+1
058:642  x2=i*77+8
059:642  LINE(x1,155)-(x2,180),6,bf
060:818  LINE (x2+3,157)-(x2+67,178),0,bf
061:393 NEXT i
062:000
063:000
064:543 REM
065:798 REM    Hier werden die Gadgets gezeichnet
066:543 REM
067:543 REM
068:458 REM     Exit - Gadget
069:543 REM
070:018 LINE (570,161)-(590,174),7,bf
071:058 LINE(585,165)-(588,170),4,bf
072:956 LINE (581,167)-(587,166),0
073:604 LINE (587,168)-(587,169),0
074:543 REM
075:666 REM     Disketten - Gadget
076:543 REM
077:205 LINE (492,162)-(517,173),2,bf  :REM Diskette
078:158 LINE (513,163)-(515,164),1,bf  :REM Screibschutz
079:456 LINE (498,168)-(512,173),6,bf  :REM Metall
080:460 LINE (494,162)-(513,166),4,bf  :REM Aufkleber
081:000
082:543 REM
083:930 REM  Drucker - Gadget
084:543 REM
085:307 LINE (412,170)-(442,173),3,bf     :REM Drucker(1)
086:069 LINE (412,168)-(442,170),6,bf     :REM        (2)
087:223 LINE (414,166)-(440,168),2,bf     :REM        (3)
088:496 LINE (416,162)-(438,166),1,bf     :REM Papier
089:067 LINE (410,168)-(412,170),1,bf     :REM Rolle 1
090:268 LINE (442,168)-(444,170),1,bf     :REM Rolle 2
091:695 PSET (418,163),0:PSET (436,163),0 :REM
092:136 PSET (418,165),0:PSET (436,165),0 :REM   Perforation
093:543 REM
094:994 REM    Kopf - Gadget
095:543 REM
096:000
097:066 LINE(339,162)-(363,174),2,bf      :REM kopf
098:407 LINE(343,164)-(347,165),6,bf      :REM auge
099:807 LINE(355,164)-(359,165),6,bf      :REM auge
100:023 LINE(351,165)-(351,166),3         :REM nase
101:082 LINE(350,167)-(352,168),3,bf
102:604 LINE(343,170)-(359,171),0,bf      :REM mund
103:821 LINE(343,172)-(359,172),1
104:241 FOR i=0 TO 10 STEP 5
105:893   LINE(345+i,170)-(347+i,170),1
106:741   PSET(346+i,171),1
107:444   LINE(345+i,172)-(347+i,172),0
108:160 NEXT
109:543 REM
110:466 REM    Eingabe - Gadget
111:543 REM
112:000
113:794 LINE (259,163)-(271,168),3,bf
114:146 LINE (271,164)-(279,167),2,bf
115:842 LINE (275,167)-(278,172),2,bf
116:602 LINE (262,174)-(267,176),1,bf
117:010 LINE (269,173)-(274,175),1,bf
118:810 LINE (276,172)-(281,174),1,bf
119:802 LINE (283,171)-(288,173),1,bf
120:543 REM
121:594 REM     Sicht - Gadget
122:543 REM
123:000
124:794 LINE (178,161)-(214,174),2,bf
125:301 FOR b=0 TO 7
126:230   LINE (182,163+b)-(210,163+b),b
127:386 NEXT b
128:385 FOR b= 15 TO 40 STEP 5
129:618   LINE (168+b,172)-(170+b,173),0,bf
130:386 NEXT b
131:543 REM
132:514 REM    Aenderungs - Gadget
133:543 REM
134:000
135:245 LINE (94,174)-(107,174),1
136:861 LINE           -(110 ,172),1
137:133 LINE           -(112,174),1
138:901 LINE           -(142,174),1
139:298 LINE (106,163)-(114,170),2,bf
140:002 LINE (115,166)-(137,167),6,bf
141:543 REM
142:722 REM     Vor-Zurueck - Gadget
143:543 REM
144:433 a=3
145:445 LINE (31,162)-(16,167),1
146:861 LINE         -(31,173) ,1
147:741 LINE (36,162)-(21,167) ,1
148:501 LINE         -(36,173) ,1
149:205 LINE (50,162)-(65,167) ,1
150:245 LINE         -(50,173) ,1
151:501 LINE (55,162)-(70,167) ,1
152:759 LINE         -(55,173) ,1  :REM      <==
153:698 LINE (41,155)-(45,179),6,bf
154:100 LINE (43,155)-(43,179),0
155:039 '
156:868 ' Command-Text-Rahmen
157:039 '
158:834 LINE (200,194)-(430,212),6,bf
159:682 LINE (205,196)-(425,210),0,bf
160:000
161:982 RETURN
162:000
163:000
164:000
165:039 '
166:703 '    Erstellen der Bildschirmmaske
167:039 '
168:664 Maske:
169:345 COLOR 7,5
170:000
171:467 LOCATE 26,61 :PRINT Anzahl" Datensätze";
172:946 LOCATE 3,10 :PRINT "       Name : "
173:838 LOCATE 3,50 :PRINT "Kontakt:"
174:506 LOCATE 5,10 :PRINT "    Vorname : "
175:266 LOCATE 8,10 :PRINT "    Strasse : "
176:706 LOCATE 10,10:PRINT "Postleitzahl : "
177:826 LOCATE 12,10:PRINT "        Ort : "
178:562 LOCATE 15,10 :PRINT "    Telefon : "
179:000
180:000
181:114 LINE (193,13)-(366,25),0,bf
182:514 LINE (195,14)-(364,24),2,b
183:000
184:258 LINE (193,29)-(366,41),0,bf
185:514 LINE (195,30)-(364,40),2,b
186:000
187:770 LINE (193,53)-(366,65),0,bf
188:842 LINE (195,54)-(364,64),2,b
189:000
190:330 LINE (193,69)-(239,81),0,bf
191:050 LINE (195,70)-(237,80),2,b
192:000
193:426 LINE (193,85)-(366,97),0,bf
194:170 LINE (195,86)-(364,96),2,b
195:000
196:610 LINE (193,109)-(366,121),0,bf
197:922 LINE (195,110)-(364,120),2,b
198:000
199:378 LINE (450,29)-(600,121),0,bf
200:306 LINE (452,30)-(598,120),2,b
201:000
202:982 RETURN
203:000
204:000
205:543 REM
206:039 REM  Hier beginnt die Mouseabfrage
207:543 REM
208:000
209:016 choose:
210:345 COLOR 7,5
211:408 Anzahl=LOF(1)/104
212:467 LOCATE 26,61 :PRINT Anzahl" Datensätze";
213:951 LOCATE 26,5 :PRINT nr;". Datensatz";
214:950 WHILE MOUSE(0)=0
215:483 x= MOUSE (1)
216:997 y= MOUSE (2)
217:316 COLOR 1,0
218:886 LOCATE 26,30
219:192 IF y>156 AND y<180 THEN
220:276   ch=INT((x-12)/77)+2
221:123   IF x<40 THEN ch=1
222:846   IF ch=1 AND such=0 THEN PRINT"   Datensatz zurück "
223:530   IF ch=1 AND such=1 THEN PRINT" Suchmodus verlassen"
224:010   IF ch=2 THEN PRINT"   Datensatz weiter "
225:874   IF ch=3 THEN PRINT"   Datensatz ändern "
226:418   IF ch=4 THEN PRINT"   Datensatz suchen "
227:778   IF ch=5 THEN PRINT"  Datensatz eingeben"
228:602   IF ch=6 THEN PRINT"     Nummer sprechen  "
229:338   IF ch=7 THEN PRINT"     Adresse Drucken  "
230:306   IF ch=8 THEN PRINT"   Datenfile löschen "
231:618   IF ch=9  THEN PRINT"      Raus hier !!     "
232:000
233:091   ELSE
234:506   PRINT"    --    --    --  "
235:216   END IF
236:196 WEND
237:000
238:491 x=MOUSE(5)
239:005 y=MOUSE(6)
240:808 IF y>160 AND y<180 THEN
241:276    ch=INT((x-12)/77)+2
242:466    IF x<40 THEN ch=1 : such = 0
243:091 ELSE
```

```
244:994     ch=0
245:863     GOTO choose
246:216 END IF
247:948 WHILE MOUSE(0)<0
248:196 WEND
249:000
250:982 RETURN
251:000
252:000
253:012 rueck:
254:683 nr=nr-1
255:795 IF nr<=0 THEN nr=1
256:558 GET #1,nr
257:492 GOSUB uebertragen
258:258 GOSUB out
259:948 WHILE MOUSE(0)<0
260:196 WEND
261:982 RETURN
262:000
263:226 vor:
264:679 nr=nr+1
265:992 IF nr>Anzahl THEN nr=Anzahl
266:558 GET #1,nr
267:492 GOSUB uebertragen
268:258 GOSUB out
269:948 WHILE MOUSE(0)<0
270:196 WEND
271:000
272:982 RETURN
273:000
274:736 change:
275:000
276:340 COLOR 7,0
277:930 change=1:GOSUB out
278:316 COLOR 1,0
279:102 x=26
280:599 change=1:dummy$=na$:y=3:ln=20:GOSUB getstring
281:804 IF in$<>"" THEN na$=in$
282:894 LOCATE 3,26:PRINT na$
283:871 change=1:dummy$=vn$:y=5:GOSUB getstring
284:732 IF in$<>"" THEN vn$=in$
285:144 LOCATE 5,26:PRINT vn$
286:175 change=1:dummy$=st$:y=8:GOSUB getstring
287:732 IF in$<>"" THEN st$=in$
288:432 LOCATE 8,26:PRINT st$
289:215 change=1:dummy$=pl$:y=10:ln=4:GOSUB getstring
290:284 IF in$<>"" THEN pl$=in$
291:012 LOCATE 10,26:PRINT pl$
292:031 change=1:dummy$=ot$:y=12:ln=20:GOSUB getstring
293:476 IF in$<>"" THEN ot$=in$
294:216 LOCATE 12,26:PRINT ot$
295:527 change=1:dummy$=tl$:y=15:GOSUB getstring
296:540 IF in$<>"" THEN tl$=in$
297:508 LOCATE 15,26:PRINT tl$
298:760 GOSUB schreib
299:982 RETURN
300:000
301:966 watch:
302:316 COLOR 1,0
303:583 GOSUB Loeschmaske
304:554 LOCATE 26,30:PRINT "  Kriterium wählen  "
305:950 WHILE MOUSE(0)=0
306:483  x=MOUSE(1)
307:997  y=MOUSE(2)
308:200   IF x>195 AND x<364 THEN
309:915     wahl=INT(y/8)+1
310:216   END IF
311:196 WEND
312:491  x=MOUSE(5)
313:005  y=MOUSE(6)
314:200   IF x>195 AND x<364 THEN
315:915     wahl=INT(y/8)+1
316:091   ELSE
317:646     GOTO watch
318:216   END IF
319:000
320:000
321:839 IF wahl = 3  THEN x=26:y=3:ln=20:w=1
322:495 IF wahl = 5  THEN x=26:y=5:ln=20:w=1
323:479 IF wahl = 8  THEN x=26:y=8:ln=20:w=1
324:791 IF wahl = 10 THEN x=26:y=10:ln=4:w=1
325:567 IF wahl = 12 THEN x=26:y=12:ln=20:w=1
326:463 IF wahl = 15 THEN x=26:y=15:ln=20:w=1
327:000
328:990 IF w=0 THEN GOTO watch
329:518 w=0
330:000
331:204 GOSUB getstring : sr$=in$
332:122 nr=0
333:495 GOSUB lese
334:753 dummy=MOUSE(0)
335:982 RETURN
336:000
337:000
338:128 neu:
339:194 GOSUB in:
340:311 neu=1
341:650 GOSUB neuschreib:
342:982 RETURN
343:000
344:000
345:376 Talk:
346:513 FOR s=1 TO LEN(tl$)
347:539     y$=MID$(tl$,s,1)
348:121     GOSUB Translate
349:986     SAY Talk$
350:403 NEXT s
351:753 dummy=MOUSE(0)
352:982 RETURN
353:000
354:000
355:236 Translate:
356:386 Talk$=" "
357:406 IF y$="0" THEN Talk$="NUHL."
358:274 IF y$="1" THEN Talk$="AYNS."
359:090 IF y$="2" THEN Talk$="TSWAY."
360:338 IF y$="3" THEN Talk$="DRXAY."
361:614 IF y$="4" THEN Talk$="FIYRX."
362:502 IF y$="5" THEN Talk$="FIHAXNF."
363:546 IF y$="6" THEN Talk$="SEHKS."
364:934 IF y$="7" THEN Talk$="SIYBEHN."
365:606 IF y$="8" THEN Talk$="AH/CT."
366:750 IF y$="9" THEN Talk$="NOYN."
367:512 IF y$=" " OR y$="/" THEN FOR p=1 TO 100:NEXT p
368:982 RETURN
369:000
370:000
371:100 Dru:
372:511 CLOSE 1
373:210 GOSUB cut
374:000
375:454 LPRINT vn$" "na$
376:736 LPRINT st$
377:704 LPRINT pl$" "ot$
378:900 LPRINT
379:724 LPRINT tl$
380:900 LPRINT
381:900 LPRINT
382:201 GOSUB oeffne
383:982 RETURN
384:000
385:000
386:630 loesch:
387:320 COLOR 2,0
388:330 LOCATE 26,32:PRINT "  Ganz sicher ???  "
389:022 BEEP
390:711 FOR pause=1 TO 3000:NEXT pause
391:586 LOCATE 26,30:PRINT "Noch einmal anwählen"
392:711 FOR pause=1 TO 3000:NEXT pause
393:514 LOCATE 26,30:PRINT "                    "
394:983 GOSUB choose
395:766 IF ch<>8 THEN RETURN
396:511 CLOSE 1
397:630 KILL "df0:telefon.dat"
398:201 GOSUB oeffne
399:982 RETURN
400:000
401:000
402:396 raus:
403:503 CLS
404:511 CLOSE 1
405:500 END
406:000
407:000
408:968 Loeschmaske:
409:316 COLOR 1,0
410:474 a$="                    "
411:374 LOCATE 3,26 :PRINT a$
412:470 LOCATE 5,26 :PRINT a$
413:614 LOCATE 8,26 :PRINT a$
414:114 LOCATE 10,26:PRINT "    "
415:014 LOCATE 12,26 :PRINT a$
416:158 LOCATE 15,26 :PRINT a$
417:650 LINE (454,32)-(596,118),0,bf
418:982 RETURN
419:000
420:000
421:000
422:506 in:
423:583 GOSUB Loeschmaske
424:336 COLOR 6,0
425:616 x=26:ln=20
426:604 y=3:GOSUB getstring:na$=in$
427:748 y=5:GOSUB getstring:vn$=in$
428:572 y=8:GOSUB getstring:st$=in$
429:848 y=10:ln=4:GOSUB getstring:pl$=in$:ln=20
430:876 y=12:GOSUB getstring:ot$=in$
431:764 y=15:GOSUB getstring:tl$=in$
432:753 dummy=MOUSE(0)
433:316 COLOR 1,0
434:258 GOSUB out
435:982 RETURN
436:000
437:000
438:102 cut:
439:784 FOR d=1 TO 20
440:708  IF ASC(MID$(vn$,d,1))<>32 THEN NEXT d
441:225 vn$=LEFT$(vn$,d)
442:982 RETURN
443:000
444:522 neuschreib:
445:752   FOR f=1 TO Anzahl
446:730     GET #1,f
447:270   IF n$<na$ THEN NEXT f
448:144   nr=f
449:763   FOR f=Anzahl TO nr STEP -1
450:730     GET #1,f
```

```
    PUT #1,f+1
  NEXT f
neu = 0

schreib:

LSET n$=na$
LSET v$=vn$
LSET s$=st$
LSET p$=pl$
LSET o$=ot$
LSET t$=tl$
PUT #1,nr
LOCATE 26,30:PRINT "Datensatz übernommen"
FOR pause = 1 TO 2000:NEXT
RETURN


out:
GOSUB Loeschmaske
IF change=1 THEN COLOR 7,0:change=0
LOCATE 3,26 :PRINT na$
LOCATE 5,26 :PRINT vn$
LOCATE 8,26 :PRINT st$
LOCATE 10,26:PRINT pl$
LOCATE 12,26 :PRINT ot$
LOCATE 15,26 :PRINT tl$
GOSUB Bild
RETURN


uebertragen:
na$=n$
vn$=v$
st$=s$
pl$=p$
ot$=o$
tl$=t$
RETURN



Bild:
 tel = ASC ( LEFT$ (t$,1))
 IF tel > 57 OR tel < 48 THEN
   GOSUB Briefumschlag
 ELSE
   GOSUB telefon
 END IF
RETURN

Briefumschlag:
LINE (460,40)-(590,90),1,bf
LINE (460,40)-(590,90),0
LINE (460,90)-(590,40),0
RETURN

telefon:
 READ xpos,ypos
 PSET(xpos+454,ypos+32),1

FOR d=1 TO 32
 READ xpos
 READ ypos
 LINE -(xpos+454,ypos+32),1
NEXT
PAINT (510,40),1
RESTORE

RETURN

DATA 71,86,69,83,64,84,65,82,61,82,52,78,55,78,65,75
DATA 45,75,40,70,52,60,61,60,67,63,68,65,77,60
DATA 81,55,86,36,79,20,70,13,70,15,62,20,52,20
DATA 40,10,47, 5,60, 4,75, 9,88,18,95,27,95,45,91,53
DATA 80,67,70,72,65,75

lese:
WHILE Anzahl-nr>0
 nr=nr+1
 GET#1,nr
 IF wahl=3 THEN
  IF UCASE$(sr$)<>UCASE$(LEFT$(n$,LEN(sr$))) THEN GOT
O lese
 END IF
 IF wahl=5 THEN
  IF UCASE$(sr$)<>UCASE$(LEFT$(v$,LEN(sr$))) THEN GOT
O lese
 END IF
 IF wahl=8 THEN
  IF UCASE$(sr$)<>UCASE$(LEFT$(s$,LEN(sr$))) THEN GOT
O lese
 END IF
 IF wahl=10 THEN
  IF UCASE$(sr$)<>UCASE$(LEFT$(p$,LEN(sr$))) THEN GOT
O lese
 END IF
 IF wahl=12 THEN
  IF UCASE$(sr$)<>UCASE$(LEFT$(o$,LEN(sr$))) THEN GOT
O lese
 END IF
 IF wahl=15 THEN
  IF INSTR(1,t$,sr$)=0 THEN GOTO lese
 END IF

 GOSUB uebertragen
 Nummer=nr
 GOSUB out
Marke3:
 such=1
 GOSUB choose
  IF ch=1 THEN RETURN

 GOSUB Auswahl
 IF w<>1 THEN
  COLOR 1,0
  LOCATE 26,30:PRINT " Suchmodus verlassen ! "
  BEEP
  FOR pause=1 TO 3000:NEXT pause
  LOCATE 26,30:PRINT "                        "

  GOTO Marke3
  END IF
 w=0
WEND
nr=Nummer
RETURN


Auswahl:
 IF ch=6 THEN w=1:GOSUB Talk
 IF ch=7 THEN w=1:GOSUB Dru
 IF ch=10 THEN w=1:GOSUB raus
 IF ch=2 THEN w=1
RETURN



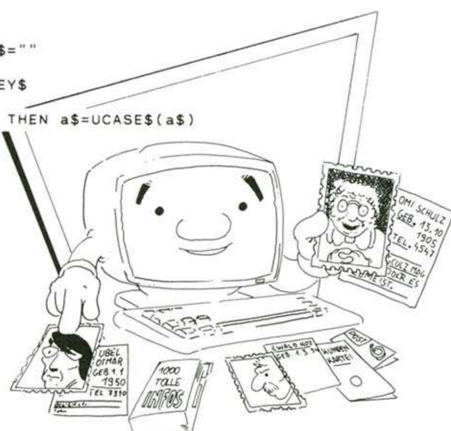
getstring:
i=0:in$="":a$=""
Markel:
LOCATE y,x+i

IF change=0 THEN
                   IF i<ln-1 THEN PRINT "_ ";
                   FOR p=1 TO ln-i-2
                    PRINT " ";
                   NEXT p
              ELSE
                   COLOR 1,0
                   PRINT LEFT$(dummy$,1);
                   change =0
              END IF
LOCATE y,x+i

 up=1:GOSUB Taste:
 IF ASC(a$)=13 THEN
  LOCATE y,x+i
  IF i<ln THEN PRINT " "
  RETURN

 END IF
 IF ASC(a$)=8 THEN
   IF LEN(in$)=>1 THEN
    in$=LEFT$(in$,LEN(in$)-1)
    a$=""
    i=i-1
   ELSE
    d=1
   END IF
 END IF
 IF i<ln THEN PRINT a$
 in$=in$+a$
 IF a$<>"" THEN i=i+1
 IF i>ln THEN
             i=i-1
             IF a$<>"" THEN in$=LEFT$(in$,LEN(in$)-1)
 END IF
 IF d=1 THEN d=0:LINE (195,y*8-10)-(195,y*8),2
GOTO Markel

Taste:
a$=""
WHILE a$=""
 SLEEP
 a$=INKEY$
WEND
IF up=0 THEN a$=UCASE$(a$)
up=0
RETURN
```



Listing 3: Dieses Adressenverwaltungsprogramm stellt die bisher erlernten und angewandten Möglichkeiten der Unterprogrammtechnik mit Hilfe von GOSUB-Anweisungen vor

Garry Glendown

SprEd – das Sprite-Editor-Projekt

Teil 4: Etwas fürs Auge

Diesmal wollen wir uns einer kleinen Nebensächlichkeit widmen: der Infoseite über das Programm SprEd. Und damit es auch wirklich gut aussieht, geben wir uns natürlich nicht nur mit einfachen Buchstaben zufrieden – unser Projekt muß schon eine ansprechende Grafik beinhalten.

Doch wie es Grafiken zumeist an sich haben, ist auch die unsere keine der kleinsten. Bei 234 mal 132 Pixel ergibt sich für den erzeugten Source-Code eine Länge von etwa 30 kByte...

Ursprünglich wollten wir deshalb diese Datenwüste als Hexdump abdrucken, doch ergaben sich leider einige Schwierigkeiten bei der anschließenden Umwandlung, so daß wir auf den Source-Code leider nicht verzichten konnten – man möge uns verzeihen...

SprEd – es geht weiter

Nach der (zugegeben mühevollen) Eingabe des Listings 'Title.c' muß das File 'part2.c' aus unserer Ausgabe 3/90 (Seite 108) noch ein bißchen modifiziert werden. Löschen Sie dazu die Zeilen 61 und 62

61: DisplayAbout()
62: {}

oder dokumentieren Sie sie einfach aus (/*...*/).

Danach kann man mit dem neuen Makefile SprEd compilieren.

In den nächsten Ausgaben wenden wir uns dem Farbrequester und der Animation der einzelnen Sequenzen zu. Bis zum nächsten Mal also!

(br)

```
# Alle Includes wurden mit
der Option '+hlib:all.pre'
vorcompiliert
OBJ = spred.o part2.o
part3.0
        title.o
        .c.o:
        cc $*.c +ilib:all.pre
        spred:
        $(OBJ)
        ln +q +cd $(OBJ) -la -
        lm -lc
```

Wichtig!
Das Programm SprEd benötigt auf jeden Fall die Arp-Library. Diese ist Public Domain und darf deswegen nicht in unserem Magazin abgedruckt werden. Wenn Sie SprEd lauffähig machen wollen, sollten Sie sich zuerst die Arp-Library besorgen. Sie befindet sich unter anderem auf der Fishdisk 123 und auf der Panorama 27c. Bei eventuellen Fragen wenden Sie sich bitte an einen PD-Vertreiber.

Listings

```
/******************************************************\
 ***                  Title.c                        ***
 ***       written in 1990 by Garry Glendown         ***
 ***            (c) 1990 by DMV-Verlag               ***
 ***            Sprache: C (Aztec 3.6)               ***
\******************************************************/

extern struct Screen *scr;

static USHORT ImageData1[] = {
   0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,
   0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,
   0xF818,0x0100,0x0180,0x1800,0x0180,0x1800,0x0180,
   0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,
   0x0180,0x1800,0x0100,0x0180,0x19FF,0x8180,0x19FF,
   0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,
   0x19FF,0x8180,0x19FF,0x8100,0x0000,0x1801,0x8000,
   0x1801,0x8000,0x1801,0x8000,0x1801,0x8000,0x1801,
   0x8000,0x1801,0x8000,0x1801,0x8000,0x1FF8,0x1801,
   0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,
   0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9F00,0x0018,
   0x0001,0x8018,0x0000,0x8008,0x0000,0x8008,0x0000,
   0x8008,0x0000,0x8018,0x0001,0x8018,0x0001,0x8000,
   0x0019,0xFF81,0x8019,0xF800,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x0019,0xFF81,0x8019,0xFF81,
   0x8000,0x0018,0x0180,0x0018,0x0200,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x0048,0x0180,0x0018,
   0x0180,0x0000,0x3818,0x019F,0xF818,0x0180,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x0000,0x00C8,0x019F,
   0xF818,0x019F,0xF800,0x1800,0x0180,0x1800,0x007F,
   0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC0,
   0x0180,0x1800,0x0180,0x1800,0x19FF,0x8180,0x19FF,
   0x8040,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
   0x03CF,0x8180,0x19FF,0x8180,0x1900,0x1801,0x8000,
   0x1801,0x8040,0x0000,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x03C1,0x8000,0x1801,0x8000,0x1800,0x1801,
   0x9FF8,0x1801,0x9040,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x03C1,0x9FF8,0x1801,0x9FF8,0x1800,
   0x0001,0x8018,0x0001,0x8040,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x03C1,0x8018,0x0001,0x8018,
   0x0000,0x3F81,0x8019,0xFF81,0x8040,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x03C1,0x8019,0xFF81,
   0x8019,0xFF00,0x0180,0x0018,0x0180,0x0040,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x0000,0x03C0,0x0018,
   0x0180,0x0018,0x0100,0x019F,0xF818,0x019F,0xF040,
   0x07E6,0x0000,0x0000,0x01FF,0xFF80,0x07E0,0x03CF,
   0xF818,0x019F,0xF818,0x0100,0x0180,0x1800,0x0180,
   0x1840,0x181E,0x0000,0x0000,0x0078,0x0780,0x01E0,
   0x03C0,0x1800,0x0180,0x1800,0x0100,0x0180,0x19FF,
   0x8180,0x1040,0x7806,0x0000,0x0000,0x0078,0x0180,
   0x01E0,0x03C0,0x19FF,0x8180,0x19FF,0x8100,0x0000,
   0x1801,0x8000,0x1840,0x7800,0x0000,0x0000,0x0078,
   0x0000,0x01E0,0x03C0,0x1801,0x8000,0x1801,0x8000,
   0x1FF8,0x1801,0x9FF8,0x1040,0x7E00,0x01F9,0xF87E,
   0x7878,0x0000,0x7FE0,0x03C8,0x1801,0x9FF8,0x1801,
   0x9F00,0x0018,0x0001,0x8018,0x0040,0x1FE0,0x007E,
   0x7E1E,0x7878,0x0601,0xE1E0,0x03C8,0x0001,0x8018,
   0x0001,0x8000,0x0019,0xFF81,0x8019,0xF040,0x07F8,
```

Listing: Title.c

```
   0x0078,0x1E1F,0x807F,0xFE07,0x81E0,0x03C9,0xFF81,
   0x8019,0xFF81,0x8000,0x0018,0x0180,0x0018,0x0040,
   0x007E,0x0078,0x1E1E,0x0078,0x0607,0x81E0,0x03C8,
   0x0180,0x0018,0x0180,0x0000,0x3818,0x019F,0xF818,
   0x0040,0x001F,0x8078,0x1E1E,0x0078,0x0007,0x81E0,
   0x03C8,0x019F,0xF818,0x019F,0xF800,0x1800,0x0180,
   0x1800,0x0040,0x0007,0x8078,0x1E1E,0x0078,0x0007,
   0x81E0,0x03C0,0x0180,0x1800,0x0180,0x1800,0x19FF,
   0x8180,0x19FF,0x8040,0x6007,0x8078,0x1E1E,0x0078,
   0x0187,0x81E0,0x03CF,0x8180,0x19FF,0x8180,0x1900,
   0x1801,0x8000,0x1801,0x8040,0x781E,0x007E,0x781E,
   0x0078,0x0781,0xE1E0,0x03C1,0x8000,0x1801,0x8000,
   0x1800,0x1801,0x9FF8,0x1801,0x9040,0x67E0,0x007F,
   0xE07F,0x81FF,0xFF80,0x7FF8,0x03C1,0x9FF8,0x1801,
   0x9FF8,0x1800,0x0001,0x8018,0x0001,0x8040,0x0000,
   0x0078,0x0000,0x0000,0x0000,0x0000,0x03C1,0x8018,
   0x0001,0x8018,0x0000,0x3F81,0x8019,0xFF81,0x8040,
   0x0000,0x0078,0x0000,0x0000,0x0000,0x0000,0x03C1,
   0x8019,0xFF81,0x8019,0xFF00,0x0180,0x0018,0x0180,
   0x0040,0x0000,0x0078,0x0000,0x0000,0x0000,0x0000,
   0x03C0,0x0018,0x0180,0x0018,0x0100,0x019F,0xF818,
   0x019F,0xF040,0x0000,0x01FE,0x0000,0x0000,0x0000,
   0x0000,0x03CF,0xF818,0x019F,0xF818,0x0100,0x0180,
   0x1800,0x0180,0x1840,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x03C0,0x1800,0x0180,0x1800,0x0100,
   0x0180,0x19FF,0x8180,0x1040,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x03C0,0x19FF,0x8180,0x19FF,
   0x8100,0x0000,0x1801,0x8000,0x1840,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x03C0,0x1801,0x8000,
   0x1801,0x8000,0x1FF8,0x1801,0x9FF8,0x1040,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x0000,0x03C8,0x1801,
   0x9FF8,0x1801,0x9F00,0x0018,0x0001,0x8018,0x0040,
   0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x03C8,
   0x0001,0x8018,0x0001,0x8000,0x0019,0xFF81,0x8019,
   0xF1FF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
   0xFCC9,0xFF81,0x8019,0xFF81,0x8000,0x0018,0x0180,
   0x0018,0x03FF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
   0xFFFF,0xFF08,0x0180,0x0018,0x0180,0x0000,0x3818,
   0x019F,0xF818,0x0000,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x0018,0x019F,0xF818,0x019F,0xF800,
   0x1800,0x0180,0x1800,0x0000,0x0000,0x0000,0x0000,
   0x0000,0x0000,0x0000,0x0000,0x0180,0x1800,0x0180,
   0x1800,0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,
   0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
   0x8180,0x1900,0x1801,0x8000,0x1801,0x8000,0x1801,
   0x8000,0x1801,0x8000,0x1801,0x8000,0x1801,0x8000,
   0x1801,0x8000,0x1800,0x1801,0x9FF8,0x1801,0x9FF8,
   0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,
   0x9FF8,0x1801,0x9FF8,0x1800,0x0001,0x8018,0x0001,
   0x8018,0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,
   0x0001,0x8018,0x0001,0x8018,0x0000,0x3F81,0x8019,
   0xFF81,0x8019,0x0001,0x8019,0xF001,0x8000,0x7F00,
   0x0000,0x0781,0x8019,0xFF81,0x8019,0xFF00,0x0180,
   0x0018,0x0180,0x0010,0x0000,0x0018,0x0000,0x0000,
   0x0000,0x0000,0x0180,0x0018,0x0180,0x0018,0x0100,
   0x019F,0xF818,0x019F,0xF800,0xFFC0,0x1818,0x0F07,
   0xC3FC,0x00FF,0x003F,0xC09F,0xF818,0x019F,0xF818,
   0x0100,0x0180,0x1800,0x0180,0x180F,0x003C,0x0800,
   0x3F00,0x0F3F,0x03CF,0xC0F0,0xF000,0x1800,0x0180,
   0x1800,0x0100,0x0180,0x19FF,0x8180,0x1930,0x0003,
   0x01FC,0xFF00,0x3C0F,0x0F03,0xC0F0,0xF000,0x19FF,
   0x8180,0x19FF,0x8100,0x0000,0x1801,0x8000,0x1830,
   0x3FC3,0x0000,0x0F00,0x3C03,0xCF00,0xF3C0,0x3C00,
```

Listing: Title.c

```
0x1801,0x8000,0x1801,0x8000,0x1FF8,0x1801,0x9FF8,
0x18C0,0xC0C0,0xC000,0x0F00,0x3C03,0xCF00,0xF3C0,
0x3C18,0x1801,0x9FF8,0x1801,0x9F00,0x0018,0x0001,
0x8018,0x00C3,0x0000,0xC001,0x0F00,0x3C03,0xCF00,
0xF3C1,0x3C18,0x0001,0x8018,0x0001,0x8000,0x0019,
0xFF81,0x8019,0xFCC3,0x0000,0xC181,0x8F01,0x3F03,
0xCFC0,0xF3C1,0x3C19,0xFF81,0x8019,0xFF81,0x8000,
0x0018,0x0180,0x0018,0x00C3,0x0000,0xC180,0x0F00,
0x0FFF,0xC3FF,0xF3C0,0x3C18,0x0180,0x0018,0x0180,
0x0000,0x3818,0x019F,0xF818,0x00C0,0xC0C0,0xC19F,
0xCF00,0x03CF,0x00F3,0xC3C1,0x3C18,0x019F,0xF818,
0x019F,0xF800,0x1800,0x0180,0x1800,0x0030,0x3F03,
0x0180,0x0F00,0x000F,0x0003,0xC3C0,0x3C00,0x0180,
0x1800,0x0180,0x1800,0x19FF,0x8180,0x19FF,0x8030,
0x0003,0x0180,0x0F01,0x803C,0x000F,0x00F0,0xF01F,
0x8180,0x19FF,0x8180,0x1900,0x1801,0x8000,0x1801,
0x800F,0x003C,0x0000,0x3FC0,0x83F0,0x00FC,0x00F0,
0xF001,0x8000,0x1801,0x8000,0x1800,0x1801,0x9FF8,
0x1801,0x9F80,0xFFC0,0x0FF8,0xFFF0,0x3F00,0x0FC0,
0x043F,0xC001,0x9FF8,0x1801,0x9FF8,0x1800,0x0001,
0x8018,0x0001,0x8000,0x0000,0x0018,0x0000,0x0000,
0x0000,0x0000,0x0001,0x8018,0x0001,0x8018,0x0000,
0x3F81,0x8019,0xFF81,0x8010,0x0000,0x0018,0x0000,
0x8001,0xE000,0x0000,0x0181,0x8019,0xFF81,0x8019,
0xFF00,0x0180,0x0018,0x0180,0x0018,0x0000,0x0018,
0x0000,0x0018,0x0000,0x0000,0x0180,0x0018,0x0180,
0x0018,0x0100,0x019F,0xF818,0x019F,0xF818,0x019F,
0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,
0x019F,0xF818,0x0100,0x0180,0x1800,0x0000,0x1800,
0x0180,0x1800,0x0180,0x1800,0x0000,0x0000,0x0000,
0x1800,0x0000,0x1800,0x0100,0x0180,0x19FF,0x8000,
0x19FF,0x8180,0x1007,0x8180,0x19FF,0x8000,0x00FF,
0x8000,0x09E0,0x0000,0x19FF,0x8100,0x0000,0x1801,
0x87C0,0x1801,0x8000,0x07F1,0x8000,0x1801,0x9FFF,
0xFC00,0x07FF,0x0007,0xFFC0,0x1801,0x8000,0x1FF8,
0x1801,0x9FF0,0x1801,0x9FF8,0x07F1,0x9FF8,0x1801,
0x9FFF,0xFF00,0x7FFF,0xF01F,0xFFC8,0x1801,0x9F00,
0x0018,0x0001,0x9FF0,0x0000,0x0000,0x07F1,0x8000,
0x0001,0x87F1,0xFFC1,0xFF07,0xFC7F,0x1FC8,0x0001,
0x8000,0x0019,0xFF81,0x1FF1,0xC000,0x0000,0xC001,
0x8000,0x0000,0x87F0,0x1FC1,0xFC01,0xFC7F,0x07C9,
0xFF81,0x8000,0x0018,0x0180,0x7FFC,0x1FF7,0xF1FC,
0x1FF0,0x1FFF,0x1FFC,0x07F0,0x1FF7,0xFC19,0xFF7F,
0xC018,0x0180,0x0000,0x3818,0x019C,0x7FFC,0x1FFF,
0xFFFF,0x1FF0,0x7FFF,0x7FFF,0x27F0,0x07F7,0xF018,
0x7F7F,0xFC18,0x019F,0xF800,0x1800,0x0181,0xFDFF,
0x07FF,0xFFFF,0x07F1,0xFFFC,0x7F7F,0x07F0,0x07F7,
0xF000,0x7F1F,0xFF00,0x0180,0x1800,0x19FF,0x8181,
0xFDFF,0x07FD,0xFF7F,0x07F1,0xFC7C,0x7FFF,0x07F3,
0x87F7,0xF1FE,0x7F07,0xFFC7,0x8180,0x1900,0x1801,
0x8001,0xFFFF,0x07F1,0xFC7F,0x07F1,0xFDFC,0x1FFF,
0x07F1,0x87F7,0xF000,0x7F00,0x7FF1,0x8000,0x1800,
0x1801,0x9FE7,0xFFFF,0xC7F1,0xFC7F,0x07F1,0xFFFC,
0x7FFF,0x07F0,0x1FF7,0xFC01,0xFF00,0x1FF1,0x9FF8,
0x1800,0x0001,0x8007,0xF07F,0xC7F1,0xFC7F,0x07F0,
0x7FF0,0x7F7F,0x07F0,0x1FC1,0xFC01,0xFC7C,0x07F1,
0x8018,0x0000,0x3F81,0x801F,0xC01F,0xF7F1,0xFC7F,
0x07F0,0x7FFC,0x7FFF,0x07F1,0xFFC9,0xFF07,0xFC7F,
0x1FF1,0x8019,0xFF00,0x0180,0x007F,0xF07F,0xFFFF,
0xFFFF,0xDFFC,0x7FFF,0xFFFF,0xDFFF,0xFF18,0x7FFF,
0xF07F,0xFFC0,0x0018,0x0100,0x019F,0xF87F,0xF07F,
0xFFFF,0xFFFF,0xDFFD,0xFFFF,0xDFDF,0xDFFF,0xFC18,
0x07FF,0x007F,0xFC0F,0xF818,0x0100,0x0180,0x1800,
0x0100,0x0000,0x0000,0x0001,0xFC1F,0xC000,0x0000,
0x0000,0x0000,0x0800,0x0000,0x1800,0x0100,0x0180,
0x1980,0x0180,0x0000,0x0000,0x0001,0xFF1F,0xC020,
0x0000,0x01FF,0x8000,0x1980,0x0180,0x19FF,0x8100,
0x0000,0x1801,0x8000,0x1801,0x8000,0x1801,0xFFFF,
0xC801,0x8000,0x1801,0x8000,0x1801,0x8000,0x1801,
0x8000,0x1FF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1800,
0x7FFF,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,
0x1801,0x9F00,0x0018,0x0001,0x8018,0x0001,0x8018,
0x0000,0x0000,0x0001,0x8018,0x0001,0x8018,0x0001,
0x8018,0x0001,0x8000,0x0019,0xFF81,0x8019,0xFF81,
0x8019,0xFF81,0x8000,0xFF81,0x8019,0xFF81,0x8019,
0xFF81,0x8019,0xFF81,0x8000,0x0018,0x0180,0x0018,
0x0180,0x0018,0x0180,0x0018,0x0180,0x0018,0x0180,
0x0018,0x0180,0x0018,0x0180,0x0000,0x3818,0x019F,
0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,
0x019F,0xF818,0x019F,0xF818,0x019F,0xF800,0x1800,
0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,
0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,
0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19F0,
0x0180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,
0x1900,0x1801,0x8000,0x1801,0x8000,0x1801,0x8000,
0x1800,0xE000,0x1801,0x8000,0x1801,0x8000,0x1801,
0x8000,0x1800,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,
0x9FF8,0x180F,0x39F8,0x1801,0x9FF8,0x1801,0x9FF8,
0x1801,0x9FF8,0x1800,0x0001,0x8018,0x0001,0x8018,
0x0001,0x8018,0x000E,0x3818,0x0001,0x8018,0x0001,
0x8018,0x0001,0x8018,0x0000,0x3F81,0x8019,0xFF81,
0x8019,0xFF81,0x8019,0xFF0E,0x3801,0xFF81,0x8019,
0xFF81,0x8019,0xFF81,0x8019,0xFF00,0x0180,0x0018,
0x0180,0x0018,0x0180,0x0018,0x000C,0xF800,0x0180,
0x0018,0x0180,0x0018,0x0180,0x0018,0x0100,0x019F,
0xF818,0x019F,0xF818,0x019F,0xF818,0x00C3,0xC038,
0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,0x0100,
0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,0x0003,
0xB0F8,0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,
0x0100,0x0180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
0x8030,0xC3E3,0x8180,0x19FF,0x8180,0x19FF,0x8180,
0x19FF,0x8100,0x0000,0x1801,0x8000,0x1801,0x8000,
0x1801,0x80F0,0x0F81,0x8000,0x1801,0x8000,0x1801,
0x8000,0x1801,0x8000,0x1FF8,0x1801,0x9FF8,0x1801,
0x9FF8,0x1801,0x9CEC,0x0E01,0x9FF8,0x1801,0x9FF8,
0x1801,0x9FF8,0x1801,0x9F00,0x0018,0x0001,0x8018,
```

Listing: Title.c

```
0x0001,0x8018,0x0001,0x80E0,0x0380,0x8018,0x0001,
0x8018,0x0001,0x8018,0x0001,0x8000,0x0019,0xFF81,
0x8019,0xFF81,0x8019,0xFF81,0x8030,0xC00E,0x0019,
0xFF81,0x8019,0xFF81,0x8019,0xFF81,0x8000,0x0018,
0x0180,0x0018,0x0180,0x0018,0x0180,0x0C03,0xF03E,
0x0018,0x0180,0x0018,0x0180,0x0018,0x0180,0x0000,
0x3818,0x019F,0xF818,0x019F,0xF818,0x019F,0xE3FF,
0x8FF8,0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,
0xF800,0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,
0x1000,0x0000,0x1800,0x0180,0x1800,0x0180,0x1800,
0x0180,0x1800,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
0x8180,0x1800,0x0000,0x19FF,0x8180,0x19FF,0x8180,
0x19FF,0x8180,0x1900,0x1801,0x8000,0x0001,0x8000,
0x1800,0x8000,0x1801,0x8000,0x1800,0x0000,0x1801,
0x8000,0x1801,0x8000,0x1800,0x1801,0x8000,0x0001,
0x9FF8,0x0000,0x0100,0x1801,0x9FF8,0x1800,0x0078,
0x1801,0x9FF8,0x1801,0x9FF8,0x1800,0x0000,0x07F8,
0x6001,0x8018,0x07F8,0x607E,0x0001,0x8018,0x0001,
0xF818,0x0001,0x8018,0x0001,0x8018,0x0000,0x3F80,
0x7807,0xE081,0x8018,0x7807,0xE01E,0x0F81,0x8019,
0xFF80,0x7819,0xFF81,0x8019,0xFF81,0x8019,0xFF00,
0x0181,0xE001,0xE080,0x0019,0xE001,0xE01E,0x0180,
0x0000,0x0180,0x7818,0x0080,0x0000,0x0000,0x0018,
0x0100,0x0191,0xE000,0x609F,0xF811,0xE000,0x601E,
0x0000,0x0000,0x0080,0x7800,0x0000,0x0000,0x0000,
0x0018,0x0100,0x0187,0x8000,0x0080,0x1807,0x8080,
0x001E,0x007F,0x87E1,0xE01F,0xF807,0xF81F,0x87E1,
0xFFE1,0xE000,0x0100,0x0187,0x81F0,0x0000,0x19E7,
0x8180,0x001E,0x01E1,0xE1E7,0xF878,0x781E,0x1E07,
0x8781,0xE1E7,0xF83F,0x8100,0x0007,0x8007,0xF800,
0x1807,0x8007,0xF81E,0x0780,0x61F8,0x79E0,0x7878,
0x0787,0x8187,0x81F8,0x7801,0x8000,0x1FE7,0x8001,
0xE038,0x1807,0x83E1,0xE01E,0x07FF,0xE1E0,0x79E0,
0x7878,0x0781,0xE787,0x81E0,0x7801,0x9F00,0x0007,
0x8001,0xE018,0x0007,0x8011,0xE01E,0x0780,0x01E0,
0x79E0,0x7878,0x0781,0xE7E7,0x81E0,0x7801,0x8000,
0x0001,0xE001,0xE000,0xFF81,0xE001,0xE09E,0x0780,
0x01E0,0x79E0,0x7878,0x0780,0x7E66,0x01E0,0x7801,
0x8000,0x0011,0xE001,0xE000,0x0181,0xE001,0xE09E,
0x0780,0x01E0,0x79E0,0x7878,0x0780,0x7E7E,0x01E0,
0x7800,0x0000,0x3818,0x7E1F,0x8078,0x0198,0x7E1F,
0x801E,0x01E1,0xE1E0,0x7878,0x781E,0x1E00,0x1818,
0x01E0,0x781F,0xF800,0x1800,0x07F8,0x0078,0x0180,
0x07F8,0x007F,0x807F,0x87F8,0x7E1F,0xFE07,0xF800,
0x1818,0x07F8,0x7E00,0x1800,0x19FE,0x0000,0x0000,
0x0180,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
0x000F,0x8000,0x0000,0x0000,0x1900,0x1801,0x8000,
0x0800,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
0x0000,0x0001,0x8000,0x1000,0x0000,0x1800,0x1801,
0x9800,0x1800,0x9FF8,0x1800,0x9F80,0x0800,0x0800,
0x0000,0x0038,0x0001,0x80E0,0x1800,0x8038,0x1800,
0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,0x0001,
0x8018,0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,
0x0000,0x3F81,0x8019,0xFF81,0x8019,0xFF81,0x8019,
0xFF81,0x8019,0xFF81,0x8019,0xFF81,0x8019,0xFF81,
0x8019,0xFF00,0x0180,0x0018,0x0180,0x0018,0x0180,
0x0018,0x0180,0x0018,0x0180,0x0018,0x0180,0x0018,
0x0180,0x0018,0x0100,0x019F,0xF818,0x019F,0xF818,
0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,
0xF818,0x019F,0xF818,0x0100,0x0180,0x1800,0x0180,
0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,
0x0180,0x1800,0x0180,0x1800,0x0100,0x0000,0x0000,
0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC0,
0xC19F,0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,
0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,
0x01C0,0xC180,0x1800,0x0180,0x1800,0x0180,0x1800,
0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,
0x1800,0x01C0,0xC180,0x19FF,0x8180,0x19FF,0x8180,
0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
0x8180,0x19FF,0x81C0,0xC000,0x1801,0x8000,0x1801,
0x8000,0x1801,0x8000,0x1801,0x8000,0x1801,0x8000,
0x1801,0x8000,0x1801,0x80C0,0xDFF8,0x1801,0x9FF8,
0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,
0x9FF8,0x1801,0x9FF8,0x1801,0x9FC0,0xC018,0x0001,
0x8018,0x0000,0x8008,0x0000,0x8008,0x0000,0x8008,
0x0000,0x8018,0x0001,0x8018,0x0001,0x80C0,0xC019,
0xFF81,0x8019,0xF800,0x0000,0x0000,0x0000,0x0000,
0x0000,0x0000,0x0019,0xFF81,0x8019,0xFF81,0x80C0,
0xC018,0x0180,0x0018,0x0355,0x5555,0x5555,0x5555,
0x5555,0x5555,0x5555,0x55C8,0x0180,0x0018,0x0180,
0x00C0,0xF818,0x019F,0xF818,0x03AA,0xAAAA,0xAAAA,
0xAAAA,0xAAAA,0xAAAA,0xAAAA,0xAAC8,0x019F,0xF818,
0x019F,0xF8C0,0xD800,0x0180,0x1800,0x017F,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC0,0x0180,
0x1800,0x0180,0x18C0,0xD9FF,0x8180,0x19FF,0x82FF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFCF,
0x8180,0x19FF,0x8180,0x19C0,0xD801,0x8000,0x1801,
0x817F,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
0xFFC1,0x8000,0x1801,0x8000,0x18C0,0xD801,0x9FF8,
0x1801,0x92FF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
0xFFFF,0xFFC1,0x9FF8,0x1801,0x9FF8,0x18C0,0xC001,
0x8018,0x0001,0x817F,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFC1,0x8018,0x0001,0x8018,0x00C0,
0xFF81,0x8019,0xFF81,0x82FF,0xFFFF,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFC1,0x8019,0xFF81,0x8019,
0xFFC0,0xC180,0x0018,0x0180,0x017F,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC0,0x0018,0x0180,
0x0018,0x01C0,0xC19F,0xF818,0x019F,0xF2FF,0xF800,
0x3FFF,0xFFFF,0xFE00,0x000F,0xF803,0xFFCF,0xF818,
0x019F,0xF818,0x01C0,0xC180,0x1800,0x0180,0x197F,
0xE000,0x3FFF,0xFFFF,0xFE00,0x000F,0xF803,0xFFC0,
0x1800,0x0180,0x1800,0x01C0,0xC180,0x19FF,0x8180,
0x12FF,0x80E0,0x3FFF,0xFFFF,0xFF80,0xF80F,0xFE03,
```

Listing: Title.c

```
0xFFC0,0x19FF,0x8180,0x19FF,0x81C0,0xC000,0x1801,
0x8000,0x197F,0x80F8,0x3FFF,0xFFFF,0xFF80,0xFE0F,
0xFE03,0xFFC0,0x1801,0x8000,0x1801,0x80C0,0xDFF8,
0x1801,0x9FF8,0x12FF,0x803F,0xFE00,0x0080,0x0080,
0xFFFF,0x8003,0xFFC8,0x1801,0x9FF8,0x1801,0x9FC0,
0xC018,0x0001,0x8018,0x017F,0x8003,0xFE00,0x0000,
0x0080,0xF83E,0x0003,0xFFC8,0x0001,0x8018,0x0001,
0x80C0,0xC019,0xFF81,0x8019,0xF2FF,0xE000,0xFF80,
0x0020,0x0080,0x0038,0x0203,0xFFC9,0xFF81,0x8019,
0xFF81,0x80C0,0xC018,0x0180,0x0018,0x017F,0xF800,
0x3F80,0xE020,0x0F80,0x0038,0x0E03,0xFFC8,0x0180,
0x0018,0x0180,0x00C0,0xF818,0x019F,0xF818,0x02FF,
0xFF80,0x0F80,0xE020,0x3F80,0xF838,0x0E03,0xFFC8,
0x019F,0xF818,0x019F,0xF8C0,0xD800,0x0180,0x1800,
0x017F,0xFFE0,0x0F80,0xE020,0x3F80,0xFFF8,0x0E03,
0xFFC0,0x0180,0x1800,0x0180,0x18C0,0xD9FF,0x8180,
0x19FF,0x82FF,0x83F8,0x0F80,0xE020,0x3F80,0xFE08,
0x0E03,0xFFCF,0x8180,0x19FF,0x8180,0x19C0,0xD801,
0x8000,0x1801,0x817F,0x80E0,0x0F80,0x0020,0x3F80,
0xF808,0x0203,0xFFC1,0x8000,0x1801,0x8000,0x18C0,
0xD801,0x9FF8,0x1801,0x92FF,0x8000,0x3F80,0x0080,
0x0E00,0x000E,0x0000,0xFFC1,0x9FF8,0x1801,0x9FF8,
0x18C0,0xC001,0x8018,0x0001,0x817F,0x8003,0xFF80,
0x0380,0x0E00,0x000F,0x8000,0xFFC1,0x8018,0x0001,
0x8018,0x00C0,0xFF81,0x8019,0xFF81,0x82FF,0xFFFF,
0xFF80,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC1,0x8019,
0xFF81,0x8019,0xFFC0,0xC180,0x0018,0x0180,0x017F,
0xFFFF,0xFF80,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC0,
0x0018,0x0180,0x0018,0x01C0,0xC19F,0xF818,0x019F,
0xF2FF,0xFFFF,0xFE00,0x3FFF,0xFFFF,0xFFFF,0xFFFF,
0xFFCF,0xF818,0x019F,0xF818,0x01C0,0xC180,0x1800,
0x0180,0x197F,0xFFFF,0xFE00,0x3FFF,0xFFFF,0xFFFF,
0xFFFF,0xFFC0,0x1800,0x0180,0x1800,0x01C0,0xC180,
0x19FF,0x8180,0x12FF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFC0,0x19FF,0x8180,0x19FF,0x81C0,
0xC000,0x1801,0x8000,0x197F,0xFFFF,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFC0,0x1801,0x8000,0x1801,
0x80C0,0xDFF8,0x1801,0x9FF8,0x12FF,0xFFFF,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC8,0x1801,0x9FF8,
0x1801,0x9FC0,0xC018,0x0001,0x8018,0x017F,0xFFFF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC8,0x0001,
0x8018,0x0001,0x80C0,0xC019,0xFF81,0x8019,0xF3FF,
0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC9,
0xFF81,0x8019,0xFF81,0x80C0,0xC018,0x0180,0x0018,
0x03FF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
0xFFC8,0x0180,0x0018,0x0180,0x00C0,0xF818,0x019F,
```

Listing: Title.c

```
0xF818,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
0x0000,0x0018,0x019F,0xF818,0x019F,0xF8C0,0xD800,
0x0180,0x1800,0x0000,0x0000,0x0000,0x0000,0x0000,
0x0000,0x0000,0x0000,0x0180,0x1800,0x0180,0x18C0,
0xD9FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,
0x19C0,0xD801,0x8000,0x1801,0x8000,0x1801,0x8000,
0x1801,0x8000,0x1801,0x8000,0x1801,0x8000,0x1801,
0x8000,0x18C0,0xD801,0x9FF8,0x1801,0x9FF8,0x1801,
0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,
0x1801,0x9FF8,0x18C0,0xC001,0x8018,0x0001,0x8018,
0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,0x0001,
0x8018,0x0001,0x8018,0x00C0,0xFF81,0x8019,0xFF81,
0x8019,0x0001,0x8019,0xF001,0x8000,0x7F00,0x0000,
0x0781,0x8019,0xFF81,0x8019,0xFFC0,0xC180,0x0018,
0x0180,0x0010,0x0000,0x0018,0x0000,0x0000,0x0000,
0x0000,0x0180,0x0018,0x0180,0x0018,0x01C0,0xC19F,
0xF818,0x019F,0xF800,0x0038,0x1818,0x00E7,0xC003,
0x8000,0xE000,0x389F,0xF818,0x019F,0xF818,0x01C0,
0xC180,0x1800,0x0180,0x1800,0xFFC3,0x8800,0x00E0,
0x00C0,0xE030,0x380F,0x0E00,0x1800,0x0180,0x1800,
0x01C0,0xC180,0x19FF,0x8180,0x190F,0xE03C,0xE1FC,
0x00E0,0x03F0,0xE0FC,0x380E,0x0E00,0x19FF,0x8180,
0x19FF,0x81C0,0xC000,0x1801,0x8000,0x180E,0x0038,
0xE000,0xF0E0,0x038C,0x30E3,0x0C3E,0xC380,0x1801,
0x8000,0x1801,0x80C0,0xDFF8,0x1801,0x9FF8,0x183E,
0x3F3B,0x3800,0x00E0,0x0380,0x30E0,0x0C38,0x0398,
0x1801,0x9FF8,0x1801,0x9FC0,0xC018,0x0001,0x8018,
0x0038,0xF8F8,0x3801,0x00E0,0x0380,0x30E0,0x0C39,
0x0398,0x0001,0x8018,0x0001,0x80C0,0xC019,0xFF81,
0x8019,0xFC38,0xE000,0x3981,0x80E1,0x00E0,0x3038,
0x0C39,0x0399,0xFF81,0x8019,0xFF81,0x80C0,0xC018,
0x0180,0x0018,0x0038,0xE000,0x3980,0x00E0,0x3000,
0x3C00,0x0C38,0x0398,0x0180,0x0018,0x0180,0x00C0,
0xF818,0x019F,0xF818,0x003B,0x3838,0x399F,0xC0E0,
0x0C30,0xFB0C,0x3C39,0x0398,0x019F,0xF818,0x019F,
0xF8C0,0xD800,0x0180,0x1800,0x00CE,0xC0F8,0xF980,
0x00E0,0x03F0,0xE0FC,0x3838,0x0380,0x0180,0x1800,
0x0180,0x18C0,0xD9FF,0x8180,0x19FF,0x800E,0x3FE0,
0xE180,0x00E1,0x8003,0xE000,0xFB0E,0x0F9F,0x8180,
0x19FF,0x8180,0x19C0,0xD801,0x8000,0x1801,0x8030,
0xE003,0xE000,0x0038,0x800F,0x8003,0xE00E,0x0E01,
0x8000,0x1801,0x8000,0x18C0,0xD801,0x9FF8,0x1801,
0x9F8F,0x003F,0x8FF8,0x000E,0x00FE,0x003F,0x84C0,
0x3E01,0x9FF8,0x1801,0x9FF8,0x18C0,0xC001,0x8018,
0x0001,0x8000,0xFFF8,0x0018,0xFFFE,0x3FE0,0x0FF8,
```

Listing: Title.c

```
    0x003F,0xF801,0x8018,0x0001,0x8018,0x00C0,0xFF81,
    0x8019,0xFF81,0x8010,0x0000,0x0018,0x0000,0x8001,
    0xE000,0x0000,0x0181,0x8019,0xFF81,0x8019,0xFFC0,
    0xC180,0x0018,0x0180,0x0018,0x0000,0x0018,0x0000,
    0x0018,0x0000,0x0000,0x0180,0x0018,0x0180,0x0018,
    0x01C0,0xC19F,0xF818,0x019F,0xF818,0x019F,0xF818,
    0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,
    0xF818,0x01C0,0xC180,0x1800,0x0000,0x1800,0x0180,
    0x1800,0x0180,0x1800,0x0000,0x0000,0x0000,0x1800,
    0x0000,0x1800,0x01C0,0xC180,0x19FF,0x8000,0x19FF,
    0x8180,0x1007,0x8180,0x19FF,0x8000,0x00FF,0x8000,
    0x09E0,0x0000,0x19FF,0x81C0,0xC000,0x1801,0x81C0,
    0x1801,0x8000,0x0071,0x8000,0x1801,0x8000,0x1C00,
    0x0007,0x0000,0x19C0,0x1801,0x80C0,0xDFF8,0x1801,
    0x8070,0x1801,0x9FF8,0x0071,0x9FF8,0x1801,0x987E,
    0x0700,0x07F8,0x7007,0xE1C8,0x1801,0x9FC0,0xC018,
    0x0001,0x8070,0x0000,0x0000,0x07F1,0x8000,0x0001,
    0x8071,0xE1C0,0x1F06,0x1C07,0x19C8,0x0001,0x80C0,
    0xC019,0xFF81,0x0671,0xC000,0x0000,0xC001,0x8000,
    0x0000,0x8070,0x01C0,0x1C00,0x1C07,0x07C9,0xFF81,
    0x80C0,0xC018,0x0180,0x061C,0x0070,0x701C,0x0070,
    0x0007,0x001C,0x0070,0x1870,0x7C19,0x8701,0xC018,
    0x0180,0x00C0,0xF818,0x019C,0x1E1C,0x1860,0x1807,
    0x1870,0x061F,0x0787,0x2070,0x0070,0x7018,0x0760,
    0x1C18,0x019F,0xF8C0,0xD800,0x0180,0x1C07,0x001E,
    0x0787,0x0070,0x1F9C,0x0707,0x0070,0x0070,0x7000,
    0x0718,0x0700,0x0180,0x18C0,0xD9FF,0x8180,0x7D87,
    0x007C,0x1F07,0x0070,0x1C1C,0x7E07,0x0073,0x8070,
    0x71FE,0x0707,0x81C7,0x8180,0x19C0,0xD801,0x8000,
    0x0007,0x0070,0x1C07,0x0070,0x1C1C,0x0187,0x0071,
    0x8070,0x7000,0x0700,0x6071,0x8000,0x18C0,0xD801,
    0x9FE0,0x7F81,0xC070,0x1C07,0x0071,0x807C,0x0787,
    0x0070,0x01F6,0x1C00,0x1F00,0x1871,0x9FF8,0x18C0,
    0xC001,0x8001,0xF061,0xC070,0x1C07,0x0070,0x1FF0,
    0x0707,0x0070,0x01C0,0x1C00,0x1C1C,0x0071,0x8018,
    0x00C0,0xFF81,0x8001,0xC000,0x7070,0x1C07,0x0070,
    0x001C,0x0007,0x0070,0x07C9,0x8700,0x7C07,0x01F1,
    0x8019,0xFFC0,0xC180,0x0000,0x7000,0x0018,0x0601,
    0xC01C,0x0001,0xE1E1,0xC000,0x1F18,0x7807,0xF018,
    0x1FC0,0x0018,0x01C0,0xC19F,0xF87F,0xF07F,0xFFFF,
    0xFFFF,0xDFFC,0x7FE1,0xDFDF,0xDFFF,0xFC18,0x07FF,
    0x007F,0xFC0F,0xF818,0x01C0,0xC180,0x1800,0x0100,
    0x0000,0x0000,0x0000,0x1C19,0xC000,0x0000,0x0000,
    0x0000,0x0800,0x0000,0x1800,0x01C0,0xC180,0x1980,
    0x0180,0x0000,0x0000,0x0000,0x0701,0xC020,0x0000,
    0x01FF,0x8000,0x1980,0x0180,0x19FF,0x81C0,0xC000,
    0x1801,0x8000,0x1801,0x8000,0x1801,0x8007,0xC801,
    0x8000,0x1801,0x8000,0x1801,0x8000,0x1801,0x80C0,
    0xDFF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1800,0x7FFF,
    0x1801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,
    0x9FC0,0xC018,0x0001,0x8018,0x0001,0x8018,0x0000,
    0x0000,0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,
    0x0001,0x80C0,0xC019,0xFF81,0x8019,0xFF81,0x8019,
    0xFF81,0x8000,0xFF81,0x8019,0xFF81,0x8019,0xFF81,
    0x8019,0xFF81,0x80C0,0xC018,0x0180,0x0018,0x0180,
    0x0018,0x0180,0x0018,0x0180,0x0018,0x0180,0x0018,
    0x0180,0x0018,0x0180,0x00C0,0xF818,0x019F,0xF818,
    0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,
    0xF818,0x019F,0xF818,0x019F,0xF8C0,0xD800,0x0180,
    0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,
    0x0180,0x1800,0x0180,0x1800,0x0180,0x18C0,0xD9FF,
    0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19F0,0x0180,
    0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19C0,
    0xD801,0x8000,0x1801,0x8000,0x1801,0x8000,0x1800,
    0x0000,0x1801,0x8000,0x1801,0x8000,0x1801,0x8000,
    0x18C0,0xD801,0x9FF8,0x1801,0x9FF8,0x1801,0x9FF8,
    0x1800,0x01F8,0x1801,0x9FF8,0x1801,0x9FF8,0x1801,
    0x9FF8,0x18C0,0xC001,0x8018,0x0001,0x8018,0x0001,
    0x8018,0x0000,0x0018,0x0001,0x8018,0x0001,0x8018,
    0x0001,0x8018,0x00C0,0xFF81,0x8019,0xFF81,0x8019,
    0xFF81,0x8019,0xFF00,0x0001,0xFF81,0x8019,0xFF81,
    0x8019,0xFF81,0x8019,0xFFC0,0xC180,0x0018,0x0180,
    0x0018,0x0180,0x0018,0x0000,0x0000,0x0180,0x0018,
    0x0180,0x0018,0x0180,0x0018,0x01C0,0xC19F,0xF818,
    0x019F,0xF818,0x019F,0xF818,0x0000,0x0000,0x019F,
    0xF818,0x019F,0xF818,0x019F,0xF818,0x01C0,0xC180,
    0x1800,0x0180,0x1800,0x0180,0x1800,0x0000,0x0000,
    0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,0x01C0,
    0xC180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,0x8000,
    0x0003,0x8180,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
    0x81C0,0xC000,0x1801,0x8000,0x1801,0x8000,0x1801,
    0x8000,0x0001,0x8000,0x1801,0x8000,0x1801,0x8000,
    0x1801,0x80C0,0xDFF8,0x1801,0x9FF8,0x1801,0x9FF8,
    0x1801,0x9C00,0x0001,0x9FF8,0x1801,0x9FF8,0x1801,
    0x9FF8,0x1801,0x9FC0,0xC018,0x0001,0x8018,0x0001,
    0x8018,0x0001,0x8000,0x0000,0x8018,0x0001,0x8018,
    0x0001,0x8018,0x0001,0x80C0,0xC019,0xFF81,0x8019,
    0xFF81,0x8019,0xFF81,0x8000,0x0000,0x0019,0xFF81,
    0x8019,0xFF81,0x8019,0xFF81,0x80C0,0xC018,0x0180,
    0x0018,0x0180,0x0018,0x0180,0x0000,0x0000,0x0018,
    0x0180,0x0018,0x0180,0x0018,0x0180,0x00C0,0xF818,
    0x019F,0xF818,0x019F,0xF818,0x019F,0xE000,0x0000,
    0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,0xF8C0,
    0xD800,0x0180,0x1800,0x0180,0x1800,0x0180,0x1000,
    0x0000,0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,
    0x18C0,0xD9FF,0x8180,0x19FF,0x8180,0x19FF,0x8180,
    0x1800,0x0000,0x19FF,0x8180,0x19FF,0x8180,0x19FF,
    0x8180,0x19C0,0xD801,0x8000,0x0001,0x8000,0x1800,
    0x8000,0x1801,0x8000,0x1800,0x0000,0x1801,0x8000,
    0x1801,0x8000,0x18C0,0xD801,0x8000,0x0001,0x9FF8,
    0x0000,0x0100,0x1801,0x9FF8,0x1800,0x0078,0x1801,
    0x9FF8,0x1801,0x9FF8,0x18C0,0xC000,0x0007,0x1C01,
    0x8018,0x0007,0x1C01,0xC001,0x8018,0x0000,0x0718,
    0x0001,0x8018,0x0001,0x8018,0x00C0,0xFF80,0x07F8,
    0x1C81,0x8018,0x07F8,0x1C61,0xCF81,0x8019,0xFF81,
    0x8719,0xFF81,0x8019,0xFF81,0x8019,0xFFC0,0xC180,
```

Listing: Title.c

```
    0x1F06,0x1C80,0x0018,0x1F06,0x1C01,0xC180,0x0000,
    0x0180,0x0718,0x0080,0x0000,0x0000,0x0018,0x01C0,
    0xC190,0x1C01,0x9C9F,0xF810,0x1C01,0x9C01,0xC000,
    0x0000,0x0080,0x0700,0x0000,0x0000,0x0000,0x0018,
    0x01C0,0xC180,0x7C00,0x7C80,0x1800,0x7C80,0x7C01,
    0xC000,0x701C,0x1C00,0x0700,0x0700,0x701C,0x001C,
    0x1C00,0x01C0,0xC180,0x71F0,0x0000,0x19E0,0x7180,
    0x0001,0xC01E,0x1E18,0x0707,0x8701,0xE1D8,0x707C,
    0x1E18,0x073F,0x81C0,0xC000,0x7000,0x0700,0x1800,
    0x7000,0x0701,0xC07D,0x9C07,0x861F,0x0707,0xD870,
    0x7670,0x7C07,0x8701,0x80C0,0xDFE0,0x7006,0x1F38,
    0x1800,0x73E6,0x1F01,0xC000,0x1C1F,0x061C,0x0707,
    0x0076,0x1870,0x701F,0x0701,0x9FC0,0xC000,0x7000,
    0x1C18,0x0000,0x7010,0x1C01,0xC07F,0xFC1C,0x061C,
    0x0707,0x0070,0x1818,0x701C,0x0701,0x80C0,0xC006,
    0x1C00,0x1C00,0xFF86,0x1C00,0x1C81,0xC070,0x001C,
    0x061C,0x0707,0x0071,0x8199,0xF01C,0x0701,0x80C0,
    0xC010,0x1C00,0x1C00,0x0180,0x1C00,0x1C81,0xC070,
    0x001C,0x061C,0x0707,0x0070,0x0181,0xC01C,0x0700,
    0x00C0,0xF819,0x81C0,0x7C07,0x0199,0x81C0,0x7C01,
    0xC61C,0x1C1C,0x0787,0x0761,0xC1F0,0x67E7,0xC01C,
    0x071F,0xF8C0,0xD800,0x7807,0xF007,0x0180,0x7807,
    0xF000,0x7180,0x7807,0x01E0,0x01D8,0x07C0,0x0707,
    0x0007,0x01C0,0x18C0,0xD9FE,0x07FF,0x007F,0x0180,
    0x07FF,0x007F,0xF07F,0xF7FF,0x7FDF,0xFFC7,0xFF0F,
    0x9F1F,0x07FF,0x7FC0,0x19C0,0xD801,0x8000,0x0800,
    0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
    0x0001,0x8000,0x1000,0x0000,0x18C0,0xD801,0x9800,
    0x1800,0x9FF8,0x1800,0x9F80,0x0800,0x0800,0x0000,
    0x0038,0x0001,0x80E0,0x1800,0x8038,0x18C0,0xC001,
    0x8018,0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,
    0x0001,0x8018,0x0001,0x8018,0x0001,0x8018,0x00C0,
    0xFF81,0x8019,0xFF81,0x8019,0xFF81,0x8019,0xFF81,
    0x8019,0xFF81,0x8019,0xFF81,0x8019,0xFF81,0x8019,
    0xFFC0,0xC180,0x0018,0x0180,0x0018,0x0180,0x0018,
    0x0180,0x0018,0x0180,0x0018,0x0180,0x0018,0x0180,
    0x0018,0x01C0,0xC19F,0xF818,0x019F,0xF818,0x019F,
    0xF818,0x019F,0xF818,0x019F,0xF818,0x019F,0xF818,
    0x019F,0xF818,0x01C0,0xC180,0x1800,0x0180,0x1800,
    0x0180,0x1800,0x0180,0x1800,0x0180,0x1800,0x0180,
    0x1800,0x0180,0x1800,0x01C0,0xFFFF,0xFFFF,0xFFFF,
    0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFFF,
    0xFFFF,0xFFFF,0xFFFF,0xFFFF,0xFFC0
};

static struct Image Image1 = {
  0,0,
  234,132,
  2,
  ImageData1,
  0x0003,0x0000,
  NULL
};

static struct Gadget Gadget1 = {
  NULL,
  0,0,
  234,132,
  GADGHBOX¦GADGHIMAGE¦GADGIMAGE,
  RELVERIFY,
  BOOLGADGET,
  (APTR)&Image1,
  NULL,
  NULL,
  NULL,
  NULL,
  NULL,
  NULL
};

#define GadgetList1 Gadget1

static struct NewWindow NewWindowStructure1 = {
  217,39,
  234,132,
  0,1,
  CLOSEWINDOW¦GADGETUP¦INACTIVEWINDOW,
  BORDERLESS¦ACTIVATE¦NOCAREREFRESH,
  &Gadget1,
  NULL,
  NULL,
  NULL,
  NULL,
  5,5,
  -1,-1,
  CUSTOMSCREEN
};

DisplayAbout()
{
struct IntuiMsg *msg,*GetMsg();
struct Window *win,*OpenWindow();
  NewWindowStructure1.Screen=scr;
  if (win=OpenWindow(&NewWindowStructure1)) {
    WaitPort(win->UserPort);
    if (msg=GetMsg(win->UserPort)) {
emptyqueue:
      ReplyMsg(msg);
      if (msg=GetMsg(win->UserPort)) goto emptyqueue;
      CloseWindow(win);
      return();
    }
  }
  else DisplayBeep();
}
```

Listing: Title.c

AMIGA-DOS-Pixel-Panorama

Was für Paris der Louvre, ist für die AMIGA DOS das Pixel-Panorama. Tatsächlich können wir Ihnen auf dieser Seite neben erlesener Raytracing-Grafik auch die ersten Bilder präsentieren, die aus den Mäusen unserer Leser geflossen sind!

Lange haben wir hier in der Redaktion auf Ihre Bilder warten müssen. Aber nun ist die Durststrecke überwunden, die zwischen dem Augenblick, da diese Zeilen geschrieben werden, und dem Moment, da Sie sie zu lesen bekommen.

Peter Schütz aus der Schweiz hat uns das Jeans-Girl zugeschickt. Die einzelnen Motive hat er mit Deluxe Paint III erstellt und dann mit Digi Paint 3 zusammengefügt.

Fantasy und Science-fiction sind die Themen, die Tobias Richter in seinen Bildern aufgreift. Er zeichnet für die orientalische Märchenstadt und den vierbeinigen Kampfroboter verantwortlich. Auch der Raytracing-Tisch, das Bild erklärt sich von selbst, stammt von Tobias Richter. Schlußendlich haben wir noch ein edles Raytracing-Bild von Richard Nicol aufgenommen. Natürlich freuen wir uns nach wie vor über Ihre selbsterstellten Bilder ganz besonders, und für jede Veröffentlichung winkt als Dankeschön ein Software-Preis im Wert von zirka 100 DM.

Mitmachen ist ganz einfach: Packen Sie Ihre Bilder auf eine Diskette, und vermerken Sie auf dem Etikett der Disk neben Ihrer kompletten Anschrift auch das Grafikformat der Bilder bzw. die Grafikprogramme, die bei der Erstellung verwendet wurden. Das Ganze stecken Sie dann in einen Umschlag und schicken es an den:

DMV-Verlag
Redaktion AMIGA DOS
Kennwort: Pixel
Postfach 250
3440 Eschwege

Ihren Einsendungen sehen wir gespannt entgegen. Einstweilen jedoch wünschen wir Ihnen viel Spaß beim Betrachten der Bilder, die wir für diese Ausgabe ausgewählt haben, aber auch bei der Erstellung neuer. (hs)



Jeans-Girl von Peter Schütz



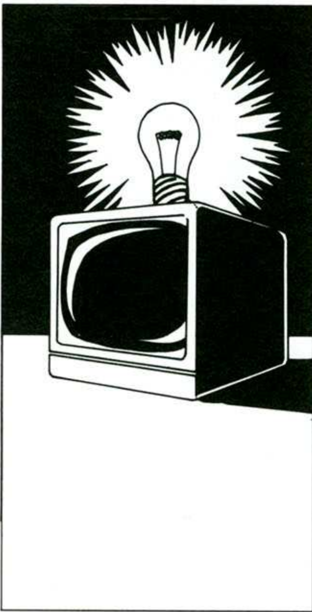
Das Land von Zomar, eine von Tobias Richters phantastischen Visionen



Science-fiction von Tobias Richter



Raytracing vom Feinsten, ein Bild von Richard Nicol



Edgar Meyzis

Tips und Tricks nicht nur für M2Amiga

Prozessieren & Exekutieren (Teil 1)

Über Rechtspraktiken des Staates X wollen wir weder heute noch im nächsten Heft berichten, wohl aber über Wege, aus einem Programm heraus andere mit einfachen Mitteln aufzustarten oder auch abzubrechen. Dabei muß es nicht unbedingt zu einem Prozeß kommen. Das aufzustartende Programm kann auch auf anderem Wege erzwingen, von der CPU bedient zu werden. Unsere Beispiele stützen sich auf M2Amiga/Modula 2.

Das Amiga-Betriebssystem stellt die Routine "Execute" zur Verfügung. In Modula ist die Schnittstelle zu "Dos.Execute" wie folgt deklariert: siehe Box 1.

Die Deklaration spricht für eine effiziente Implementierung. Bei Aktivierung der Compiler-Option "$Z-" werden sämtliche Parameter direkt in Registern (d1 bis d3) übergeben. Es wird keine übliche Prozedur mit Eintritts- und Austrittscode generiert, sondern einfach Code gemäß Listing 1 anstelle des Aufrufs eingefügt [1].

Geheimprozeß?

Die Systemroutine "Execute" entspricht nicht dem gleichlautenden CLI-Befehl. "Dos. Execute" ist dazu bestimmt, Programme zur Ausführung zu bringen, die vom CLI aus aufstartbar sind [2]. Das können sowohl die üblichen CLI-Befehle (z.B. dir, delay) als auch eigene Werke sein. "Dos.Execute" setzt eine CLI-Umgebung voraus, um stets sauber zu arbeiten. An dieser Stelle wird bereits deutlich, daß "Dos.Execute" bei der Verwendung von der Workbench aus Probleme bereiten kann. – Wir kommen in der nächsten Ausgabe darauf zurück.

Die Stärke von "Dos.Execute" liegt in der Fähigkeit, auch "BCPL-Programme" einwandfrei aufzustarten. Die originären CLI-Befehle sind in BCPL geschrieben.

"Dos.Execute" legt einen Prozeß an, ohne daß dies nach außen sichtbar wird.

Keine Exekution ohne Prozeß

Der Prozeß trägt – wie alle mit "run" kreierten Prozesse – stets den Namen "Background CLI". Es kann mehrere "Geheimprozesse" mit diesem Namen geben. Somit ist nicht sichergestellt, daß mit der Routine "Exec.FindTask" auch wirklich die Adresse des neuen Prozesses ermittelt werden kann.

Residentes "RUN"

"Dos.Execute" ist vom CLI-Befehl "run" abhängig, der den erwähnten Hintergrundprozeß einrichtet. Es hält sich noch immer hartnäckig das Gerücht, ein residentes "run" (im Sinne von Dos, nicht Exec) würde nicht genutzt werden. Mit der Version 1.3 des Betriebssystems besteht die Restriktion nicht mehr. Nahezu sämtliche CLI-Befehle können resident geladen werden. Es empfiehlt sich daher, die "StartUp-Sequence" um die Anweisung "resident c:run pure" zu ergänzen.

```
PROCEDURE Execute (commandString{1}:ADDRESS;
                   input{2} :FileHandlePtr;
                   output{3}:FileHandlePtr):LONGINT;
                   CODE -222;
```

Box 1. Procedure execute

Prozesse unterscheiden sich unter anderem durch Merkmale wie Stackgröße, Priorität, das auszuführende Programm, Ein- und Ausgabekanäle, das aktuelle Verzeichnis oder das Anhängsel "CLI-Struktur". Beim Aufruf von "Dos.Execute" befinden wir zum Beispiel nicht über die Stackgröße oder die Priorität. Das Betriebssystem geht eigene Wege und verpaßt dem neuen Hintergrundprozeß die Merkmale einer bereits resident geladenen Shell oder, falls nicht vorhanden, die des im Kickstart-ROM abgelegten CLI [3]. Daraus folgt, daß zum Beispiel die Größe des Stacks nicht unmittelbar beeinflußbar ist. Ein Aufruf von "...Execute (ADR("C:STACK 40000"),..." würde nicht zum gewünschten Erfolg führen. Nur der von "Dos.Execute" bzw. "run" angelegte, kurzlebige (!!) Prozeß würde davon profitieren, ohne je ein anderes Programm als "STACK" auszuführen. Schade!

Synchronisierte Prozesse

In einem Multitasking-System muß es möglich sein, Prozesse zu synchronisieren, um Abhängigkeiten Rechnung zu tragen – zum Beispiel, wenn ein Prozeß auf die Arbeitsergebnisse eines anderen angewiesen ist. Mit "Dos.Execute" steht eine perfekte "Rendezvous-Technik" zur Verfügung. Der Code, der einer "Execute-Anweisung" folgt, wird erst ausgeführt, wenn das aufgerufene Programm abgearbeitet und der Hintergrundprozeß beendet ist. Nach außen verhält sich "Dos.Execute" somit wie ein einfacher Prozeduraufruf. Die "Rendez-vous-Technik" läßt sich bei Bedarf austricksen, um asynchrone Prozesse anzulegen. Dazu ist der "commandString" mit "run" zu beginnen. Nun verhält sich "Dos.Execute" nahezu wie ein Programmaufstart mit "run" aus dem CLI. Der Programmteil hinter der "Execute"-Anweisung läuft parallel zum aufgestarteten Programm ab.

Hintergründige Parameter

Drei Parameter sind mit Werten zu versehen. Der "command-String" (bezeichnet genaugenommen die Adresse einer mit "null" (engl.), dem niedrigwertigsten ASCII-Zeichen, terminierten Zeichenkette) ist so zu behandeln, wie die Eingabezeile des CLI bei Aufstart eines Programms. Insbesondere sind auch die Umleitungen für Ein- und Ausgaben ("<, >, >>") zulässig. Sie werden bei der Interpretation des "command-String" erkannt und richtig umgesetzt.

Der Parameter "input" hat schon häufig falsche Erwartungen erweckt. Ein Wert ungleich "NULL" führt dazu, daß nach Abarbeitung des Programms Eingaben von der entsprechenden Datei erwartet werden. Es ist somit nicht möglich, mit "input" zu bestimmen, von welchem Datenstrom während der Ausführung (des mit Dos.Execute aufgestarteten Programms) Terminaleingaben zu lesen sind. Die Technik des Umleitens kann einspringen, sofern es ein geöffnetes "Console-Fenster" gibt und dieses dem Prozeß bekannt ist. Durch Anfügen von "<*" an den "commandString" werden Eingaben in das "Console-Fenster" zur Verarbeitung übernommen.

Beispiel: Mit "..Execute(ADR ("ReadTest <*"),..." werden die vom Programm "Read-Test" geforderten Eingaben vom aktuellen "Console-Fenster" gelesen.

Ein Problem tritt hervor: Wenn kein "Console-Fenster" dem Prozeß bekannt ist, zum Beispiel bei Aufstart von der Workbench, dann stehen wir auf dem Schlauch. Wir sind

somit praktisch auf eine CLI-Umgebung angewiesen, die sich durchaus schaffen läßt, wie wir in der nächsten Ausgabe sehen werden.

Betrachten wir den dritten und letzten Parameter, das heißt "output": Ein Wert von "NIL" führt dazu, daß Ausgaben im aktuellen Console-Fenster des aufrufenden Prozesses erfolgen. Bei Angabe eines Wertes ungleich "NIL" erfolgt die Ausgabe in den so bezeichneten Datenstrom, der sowohl ein "Console-Fenster" als auch eine Datei sein kann. Auch hier wird die Abhängigkeit von einer CLI-Umgebung sichtbar.

In die Röhre geschaut

Was ist zu tun, wenn die Ausgaben eines Programms weiter zu verarbeiten sind? Ganz einfach! Wenn Ausgaben im "NIL-Device" versickern dürfen, dann müssen sie auch in eine "PIPE" geleitet werden können. Listing 2 zeigt, wie es gemacht wird. Das Programm setzt voraus, daß der "Pipe-Handler" installiert ist. Die "StartUp-Sequence" stellt sich derartigen Aufgaben ("Mount Pipe") gerne. Obwohl die "PIPE" keine Datei ist, will sie fast wie eine solche behandelt werden. Sie ist zu öffnen und zu schließen. Der "PIPE-Handler", ein Prozeß, kann mehrere "PIPES" verwalten. In unserem Beispielprogramm hat die "PIPE" den Namen "P" erhalten. Beim Schreiben in die "PIPE" wird davon ausgegangen, daß nur eine Röhre offen ist.

Beim Experimentieren mit "PIPE" werden Sie schnell entdecken, daß sie ein vorzügliches Mittel zur Kommunikation zwischen Prozessen abgibt [4]. Sie werden auch bestätigt finden, daß "PIPES" mit "oldFile" zu öffnen sind, um uneingeschränkt nutzbar zu sein. Es werden aber auch die Grenzen von "PIPES" deutlich: Sie sind nur vier kByte "lang" und verhindern praktisch den Dialog mit dem aufgestarteten Programm. Das mangelnde "Fassungsvermögen" wirkt sich jedoch nur bei synchron ablaufenden Prozessen aus, wenn zwischendurch nicht "abgepumpt" werden kann.

Trügerisches Ergebnis

Die Systemroutine "Dos.Execute" liefert als Ergebnis null (DosFALSE) oder minus eins (DosTRUE). Mit DosFalse wird signalisiert, daß "run" zum Anlegen des Hintergrundprozesses nicht verfügbar war, weder in der von Dos geführten Liste der resident gehaltenen Programme noch im Verzeichnis "C". Folglich wurde auch nicht der Versuch unternommen, das mit Parameter eins bestimmte Programm aufzustarten.

Das Ergebnis DosTRUE bedeutet aber noch lange nicht, daß das gewünschte Programm aufgestartet werden konnte. Es ist somit nicht möglich, aus dem Ergebnis auf das Gelingen der gesamten Operation "Execute..." zu schließen. Als Programmierer übernimmt man die Verpflichtung, dafür zu sorgen, daß das aufzustartende Programm vorhanden ist und mit dem richtigen Pfad in den "commandString" eingetragen wurde. Kann man Pfad und Namen nicht schon bei der Programmierung fest eintragen, so bleibt zur Laufzeit nichts anderes übrig, als die Liste der resident geladenen Befehle zu durchsuchen, oder sich mit "Dos.Lock/Dos.UnLock" von der Existenz der Programmdatei zu überzeugen.

Über Dateioperationen des aufgestarteten Programms sind jedoch Informationen nach Auswertung des Sekundärergebnisses mit der Systemroutine "Dos.IoErr()" verfügbar. Es werden die bekannten Dos-Fehlercodes gemeldet. Mit Null wird signalisiert, daß kein Fehler auftrat.

Listing 3 zeigt, wie bei der Auswertung der beiden möglichen Ergebnisse vorgegangen werden kann. Falls "run" zur Erzeugung der Hintergrundprozesse nicht ladbar ist, so wird der selbstdefinierte Fehlercode "55" gemeldet. Sollte das Verzeichnis "X" nicht vorhanden sein, dann ermittelt "Dos.IoErr()" den Fehlercode "205" (object not found). Beachten Sie bitte, daß ein explizit aufgeführtes "run" zur Bildung eines asynchron ablaufenden Prozesses durch keinen Fehlercode erfaßt wird, dafür aber die Bemühung, das so aufzustartende Programm zu laden. Ein Fehlercode ungleich null kann aber auch durch sich anschließende Dateioperationen hervorgerufen sein.

Ausblick

In der nächsten Ausgabe werden wir die heute erworbene Erfahrung in einer CLI-Umgebung auf die Workbench übertragen und Wege aufzeigen, die Notbremse zu ziehen, das heißt, asynchrone Prozesse abzubrechen. Weiterhin werden wir synchrone Prozesse durch andere Programmiertechniken ersetzen und asynchrone Prozesse "von Hand schmieden".

(mm)

Literatur:

[1] M2Amiga Benutzerhandbuch, 1989

[2] The Amiga-Dos-Manual, 1989

[3] Das Amiga Guru-Buch, 1989

[4] Amiga ENHANCER SOFTWARE Version 1.3 bzw. neuere Begleitdokumentation zum Amiga (Amiga-Dos)

Listings

```
      lea     -86(pc),a3 ; Adresse auf commandString
      move.l  a3,d1      ; in Parameter 1
      moveq   #0,d2      ; input
      moveq   #0,d3      ; output
      move.l  -66(pc),a6 ; Basisadr. Dos-Lib.
      jsr     -222(a6)   ; Dos.Execute ausfuehren
```

Listing 1: Aufruf der Systemroutine "Dos.Execute" ohne Umweg über den Stack

```
MODULE Pipe;

FROM Dos IMPORT
        Close, Execute, FileHandlePtr, oldFile, Open;
FROM SYSTEM IMPORT
        ADR;

VAR ergebnis : LONGINT;
    fileHdl  : FileHandlePtr;

BEGIN
(* Es wird vorausgesetzt, dass in StartUp-Sequence das
   Pipe-Device angelegt wurde:
                    C:Mount PIPE: FROM DEVS:MountList
*)
  fileHdl := Open (ADR("PIPE:P"), oldFile);
  IF fileHdl # NIL THEN

     (* in die PIPE: hineinpumpen *)
     ergebnis := Execute (ADR("dir >PIPE:"), NIL, NIL);
     Close (fileHdl);

     (* und nun die PIPE: leeren *)
  ergebnis := Execute (ADR("c:type PIPE:P"), NIL, NIL);

   END;
END Pipe.
```

Listing 2: PIPES sind ein hervorragendes Mittel zur Kommunikation zwischen Prozessen

```
MODULE XO;

FROM Dos IMPORT          Execute, IoErr;
FROM SYSTEM IMPORT ADR;

CONST DosTRUE  = -1;
      DosFALSE =  0;
      NoRun    = 55; (* privater Code *)

VAR ergebnis,
     fehler   : LONGINT;
(* $Z- *)
BEGIN
   ergebnis := Execute (ADR("dir x"), NIL, NIL);
   IF ergebnis = DosTRUE THEN (* Prozess angelegt *)
     fehler   := IoErr() (* Konnte dir arbeiten ? *)
   ELSE
     fehler := NoRun (* Prozess nicht angelegt *)
   END;
END XO.
```

Listing 3: DosTRUE und IoErr () = 0 als Mindestvoraussetzung, daß alles glatt ging

Ralf Leithaus, der schon als Mitherausgeber der Public-Domain-Bücher aus dem Technik-Support-Verlag verantwortlich zeichnet, hat – diesmal in Eigenregie – ein Lehrbuch für den Umgang mit dem AmigaDOS herausgebracht. AmigaDOS für Anwender, so der Titel, ist aber nicht nur für Fortgeschrittene gedacht, nein, auch der Neuling findet einen gelungenen Einstieg in den Umgang mit den DOS-Versionen 1.2 und 1.3.

AmigaDOS für Anwender

Nach jedem Kapitel wird der gesamte Inhalt noch einmal kurz zusammengefaßt und der gelernte Stoff durch verschiedene Aufgaben abgefragt. Die Lösungen zu den einzelnen Aufgaben sind im Anhang des Buches zu finden. Diese Art der Lesereinbeziehung findet besonderen Anreiz und lockert den doch stellenweise recht trockenen Lehrstoff des Betriebssystems etwas auf. Somit kann der Leser aktiv beim Durcharbeiten des Stoffes mitarbeiten.
Nachdem grundsätzliche Begriffe geklärt wurden, werden anhand von einfachen Beispielen dem Leser einige Kopierkniffe beigebracht. Hier findet man auch vieles, was im Handbuch einfach nicht enthalten ist. Nachdem der Unterschied zwischen dem CLI und der Shell herausgestellt wurde, ist der Leser schon im nächsten Kapitel.
Auch hier wird mit diversen Beispielen der Umgang mit den Directories und Subdirectories veranschaulicht. Aber auch die logischen Devices, ein Problem, mit dem sicherlich jeder Amiganer schon einmal zu kämpfen hatte, werden schnell plausibel gemacht.
Warum kann man so einfache Befehle wie DIR oder COPY überhaupt ausführen? Fragen, über die oftmals nur gelächelt, aber dennoch keine Antwort gefunden wird, werden hier klipp und klar beantwortet. Dieses Basiswissen versucht das Buch dem Leser zu vermitteln, damit er die gesamte Funktionsweise des AmigaDOS versteht.
Auch das Prinzip des Multitasking versucht der Autor mit einigen Beispielen dem Leser näherzubringen. Ferner werden die verschiedenen Dateitypen und deren Funktionen hervorgehoben. In einem weiteren Kapitel werden dann Peripheriegeräte, wie Diskettenlaufwerke, Modem, Drucker usw., vorgestellt.
Das letzte Kapitel beschäftigt sich mit der Festplattenverwaltung. Hier wird die Installation und das Ansprechen bzw. die vollständige Nutzung einer Festplatte deutlich hervorgehoben.
Neben den schon erwähnten Lösungen zu den Aufgaben werden im Anhang zusätzlich die Fehlercodes des AmigaDOS aufgelistet.

(Jürgen Seibel/br)

Name: AmigaDOS für Anwender
Autor: Ralf Leithaus
Verlag: Technik Support
ISBN: 3-926847-09-3
Preis: 49,– DM



Ein bißchen Tuning kann nicht schaden. Das sagen sicherlich nicht nur diverse Autofahrer, sondern auch viele Amiga-Besitzer. Und genau diesen Bastlerkreis will das Buch Amiga-Hardware-Tuning ansprechen.

Amiga-Hardware-Tuning

Aber auch an diejenigen, die die Funktion der Hardware ihres Computers besser verstehen möchten, will sich dieses Buch richten. Dabei besticht das Buch auf den ersten Blick durch die gelungene Auswahl der Erweiterungen, mit denen man im Selbstbau seine "Freundin" aufrüsten kann. So werden beispielsweise eine RAM/ROM-Disk, deren Platine dem Buch beiliegt, ein Epromer, ein MIDI-Interface, ein Genlock-Interface sowie ein Sampler zum Selbstbau besprochen. Da schlägt das Bastlerherz höher. Doch die anfängliche Freude ist schnell vorbei. Spätestens wenn der Amiga kaputt in der Ecke steht, fragt man sich, was einem die Anschaffung dieses Buches eigentlich Positives gebracht hat.

Angefangen bei kleinen Fehlern wie eine ROM-Disk, die den Status 'Write-Only' trägt (oder handelt es sich hier tatsächlich um das sagenumwobene Write-Only-Memory, kurz WOM genannt?), ein Analog-Digital-Wandler, der eigentlich als Bilderzeuger im Amiga ein Digital-Analog-Wandler ist, bis hin zum fatalen Fehler im Bestückungsplan des Epromers. Dort ist das IC 74LS86 falsch gepolt, das heißt, die alles entscheidende Kerbe müßte eigentlich auf die andere Seite zeigen. Dieses kleine Mißgeschick hat zur Folge, daß sich der Epromer beim ersten Test sofort mit einer kleinen Rauchwolke in die ewigen Jagdgründe verabschiedet. Sollte man es dennoch gemerkt und das IC richtig bestückt haben, so hat man bestimmt den unerlaubten Kontakt zweier Leiterbahnen unter dem Eprom-Sockel übersehen, der bereits als Fehler auf dem Platinen-Layout anzutreffen ist.
Trotzdem uns diese Fehler gerade noch rechtzeitig auffielen, war es uns nicht möglich, den Epromer in Betrieb zu nehmen, da die auf der Diskette mitgelieferte Software völlig versagte: Bereits sofort nach Aufruf des Brenners verabschiedete sie sich ohne Anzeichen eines Fehlers und ward zur Speicherleiche.
Ganz ähnlich verhielt sich das Programm, das die Dateien zum Brennen der Eproms für die ROM-Disk erstellen sollte. Obwohl nur sechs Eproms vorgesehen waren, erstellte es neben den Dateien ROM1 bis ROM6 auch noch ROM129 und ROM130, wenn es nicht sogar mit einem 'Software Error – Task held...' den Guru meditieren ließ.
Den Test weiterer Soft- und Hardware haben wir unseren Computern erspart, da wir der Fehler überdrüssig waren und kein weiteres Risiko eingehen wollten.

Name: Amiga-Hardware-Tuning
Autoren: Uwe Gerlach und Christian Hochberger
Verlag: Markt&Technik
ISBN-Nr.: 3-89090-586-2
Preis: 98,– DM

Um alle Erfahrungen auf einen Nenner zu bringen, könnte man sagen: Gut gedacht, schlecht gemacht. Selbst einem Profi auf diesem Gebiet würde die Inbetriebnahme der Erweiterungen nur mühsam oder überhaupt nicht gelingen. Von den Problemen eines Amateurs ganz zu

schweigen. Deshalb unsere Meinung: Wer für dieses Buch 98,- DM bezahlt, ist selber schuld.

(Bernd Rudolf/br)

Der Amiga – das unbekannte Wesen. Viele Amiga-Besitzer stolpern immer dann, wenn Sie tiefer in die Programmierung des Computers einsteigen wollen, über Verständnisschwierigkeiten, die vor allem das Betriebssystem und den internen Aufbau des Amiga angehen. Die meisten auf dem Markt befindlichen Bücher sind einfach zu speziell auf Themen ausgerichtet; ein Nachschlagewerk zum 'Eben-mal-Reinschauen', das die fundamentalsten Fragen beantwortet, befindet sich jedoch kaum darunter. Was bleibt also? Für jedes Programm, das man erstellen will, muß eine Masse Literatur platztilgend neben dem Rechner liegen. Der Sybex-Verlag hat sich dieser Thematik angenommen und durch seine Autoren Garry Glendown und Roland Haas ein komplettes Nachschlagewerk erstellen lassen – das Amiga Profi-Buch.

Das Amiga Profi-Buch

Dieses Werk richtet sich vor allem an die Insider des Bereichs Amiga; dementsprechend wurde auch wenig auf die Grundsätze zur Hard- und Software eingegangen. Trotzdem zielt das Buch auf einen festen Bestandteil vom Equipment des Nicht-nur-Anwenders.

Was bietet das Profi-Buch dem Leser? Zunächst einmal komplexe Information zur Hardware des Amiga. Seine Besonderheiten, die Customchips, die Portbausteine, DMA-Technik, all das, was die Technik des Amiga ausmacht, wird unter Zuhilfenahme anschaulicher Grafiken verständlich gemacht. Dabei legen die Autoren auch Wert auf die Tatsache, daß es unterschiedliche Versionen des Amiga gibt. Soll heißen, daß hier keine Spezialisierung auf die einzelnen Typen stattfindet, sondern auch Unterschiede nebeneinander aufgezählt werden.

Als nächstes wird die CPU (der Motorola 68000) untersucht. Blockdiagramme und Grafiken über enthaltene Register bieten dem Leser die Möglichkeit, nach Lesen des Textes das Erlernte auch anhand eines Schaubildes nachzuvollziehen. Die Pin-Belegung des Prozessors sowie die Erklärung der einzelnen Signale, die an den Prozessorpins anzuliegen haben, fehlen im Buch auch nicht. Weiterhin werden dem Leser die Unterschiede des 68000 zu seinen 'Nachfolgern' 68010, 68020 und 68030 erklärt, so daß sich jeder ausmalen kann, wo bei einem eventuellen Wechsel des Prozessors die Stärken und Schwächen am meisten zum Tragen kommen.

Die Customchips werden dazu nicht nur einfach erwähnt, sondern ihr interner Aufbau ebenso präzise behandelt wie der des Hauptprozessors, dito die CIA-Bausteine. Wer schon immer etwas über die enthaltenen Timer oder die Programmierung der Ports wissen wollte, ist beim Profi-Buch genau richtig. Serieller und paralleler Port werden komplett erklärt, der Aufbau, die Pinbelegung und die anliegenden Signale sind genauestens beschrieben und trotzdem nicht nur für Techniker verständlich. Gleiches trifft auch für die Kapitel über den direkten Speicherzugriff und über die Tastatur-Decodierung zu.

Im allgemeinen bekommt man bei den Hardwarebeschreibungen stark das Gefühl, daß hier endlich einmal Wert auf Wissensübermittlung und nicht nur auf Wissensdarstellung gelegt wurde.

Die Hardware bestimmt zwar einen großen Teil des Gesamtwerkes, die Systemsoftware jedoch bildet eindeutig den Schwerpunkt. Hier merkt man, daß das Buch an Fortgeschrittene und Profis gerichtet ist: Fast alle Programmbeispiele sind in der Sprache 'C' geschrieben, viele Erklärungen sind außerdem mit Assemblerroutinen unterlegt. Kein Wunder, da sich mit diesen beiden Sprachen am effektivsten das Amiga-System einsehen läßt. Angefangen wird mit der Exec-Base, dem Grundstein zum Arbeiten mit dem Betriebssystem. Die einzelnen Vektoren der Exec-Base werden beschrieben und ihre Funktionen ausführlich dargestellt. Über die Tasks, Interrupts, Messages, Ports und Speicherzugriffe geht man langsam aber stetig in Richtung Intuition-Programmierung, einem der interessantesten, aber auch schwersten Kapitel zum Thema Amiga-Programmierung. Aber auch hier merkt der Leser schnell, daß die Autoren nicht zu den oberflächlichen Schreibern gehören, sondern ihr gesamtes Wissen dem Leser zur Verfügung stellen.

Die beiden nächsten Kapitel befassen sich mit dem sogenannten "Disk-Operating-System" des Amiga. Auch hier wird wieder ins Detail gegangen: Der Aufbau der Disketten wird anhand der Beschreibung des Root-Blocks, des Directory-, File-Header-, File-List- und Data-Blocks gezeigt. Im Anschluß daran, werden alle Library-Routinen erwähnt, dazu passend Deklarationen und Aufrufe in 'C' und Assembler mit zusätzlicher Beschreibung der benötigten Parameter. Und wem bis jetzt das Buch noch nicht geholfen hat, der findet spätestens hier eine wertvolle Unterstützung beim Programmieren.

Im Anhang finden sich noch die Befehlsausführungszeiten des Prozessors, eine alphabetische Auflistung aller Customchip-Register, die Guru-Meditation-Fehlermeldungen, die Library-Routinen nach Offset sortiert, eine Liste der Tastencodes, Beschreibung des IFF-Formates, C-Operatoren und ihre Prioritätsverteilung sowie eine Beschreibung der Programme, die auf der dem Buch beiliegenden Diskette zu finden sind.

Dieses Buch kann auf jeden Fall denen empfohlen werden, die konsequent die Programmierung des Amiga mittels systemnaher Sprachen fortführen oder ihr Wissen über die internen Vorgänge des Computers erweitern wollen.

(jb/br)

Name: Amiga Profi-Buch
Autor: Glendown, Haas
Verlag: Sybex
ISBN: 3-88745-580-0
Preis: 79,- DM

Dem Amiganer ist nur allzu bekannt, daß das AmigaDOS nicht gerade das Gelbe vom Ei ist. Durch diverse Utilities lassen sich im DOS umständliche Operationen immens vereinfachen und um ein Vielfaches beschleunigen. Das Buch 'Die besten Amiga Utilities' soll nun dem kaufwilligen Benutzer eine Orientierung im Dschungel der Utilities geben. Zugleich sollen die Amiga-Besitzer, die zugleich die im Buch vorgestellten Utilities besitzen, die gesamte Kapazität dieser Tools nutzen können, da manchen Programmen meist nur klägliche Beschreibungen beiliegen oder die Handbücher nur in englischer Sprache gehalten sind.

Die besten Amiga Utilities

In dem Buch werden 21 Dienstprogramme vorgestellt, die das Arbeiten mit dem Amiga erleichtern sollen.

Jedes Utility wird mit einem Vorwort eingeleitet, das die Funktion des jeweiligen Programms vorstellt. Im folgenden werden die einzelnen Menüs und Befehle gezeigt. Kleinere Anwendungsbeispiele sollen dem Leser den Umgang mit seinem Programm dabei einfacher gestalten.

Im Anhang findet der Leser schließlich die Bezugsadressen der vorgestellten Utilities. Hier eine Zusammenstellung der Programme, die im Buch erwähnt werden:

Autoboot-Ramdisk
Butcher V2.0
Create-A-Shape
R.C.T
Aztec 3.6a
Boot-Selector,
PowerWindows 2.5
CygnusEd Professional
Sprite Animator
Hires-Workbench
WShell
Discovery
Diskmaster
Dos-2-Dos
Sherlock
Modeler 3D
B.A.D
Quaterback
TurboPrint II
Zenon
Zing!Keys

(Jürgen Seibel/br)

Name: Die besten Amiga Utilities
Autor: Andreas Polk
Verlag: DATA Becker
ISBN: 3-89011-108-4
Preis: 39,- DM

Holt Euch zu AMIGAtionellen Preisen!!



Staubschutzhauben im New Art Design	DM 39,-
Speichererweiterung A500 intern, abschaltbar mit Uhr	DM 189,-
extern. Laufwerk mit durchgeführtem Bus, abschaltbar	DM 199,-
Citizen Swift 24 der Testsieger unter den 24 Nadel-Farbdruckern	DM 998,-

bei: Miky Wenngatz
Jägerweg 31, 8031 Gilching
Telefon 08105/24540

Fordert unseren kostenlosen Katalog an

Commodore® Ersatzteil Service

✖ Wir liefern für **Händler** und Privatanwender preiswert und prompt

✖ Rufen Sie uns an: (02331-43001) oder schreiben Sie uns:

CIK-Computertechnik • Ingo Klepsch
Berliner Straße 49b • D-5800 Hagen 7

TELEFAX: 02331-42499

UNGLAUBLICH !!

Aber wir haben wirklich über 7000 AMIGA-PUBLIC-DOMAIN !!!

kopiert auf 3,5" .. ab 2,60
kopiert auf 5,25" .. ab 1,40

5 deutsche Katalog-Disk (3,5") + die neuste TIME : DM 20,-- (VK)

diverse Super- Sonder-PD :
z.B. Return to earth Vers. 1.1 : 5,--
AVD mit VirusX 4.0 u.v.m.: 5,--

Info-Disk (3,5") ... DM 1,--

A.P.S. -electronic-
Sonnenborstel 31
D-3071 Steimbke
Tel.: 05026/1700
FAX 05026/1615

BCom Datentechnik
Chemnitzer Str. 82
D-3320 Salzgitter
Tel.: 05341/46954
FAX 05341/15061

---

10 000 AMIGA MS-DOS

Public Domain kopiert auf

2DD Markendisketten

z.B. Maxell, TDK, C=

ab **2,90**

No Name ab 2,– DM

Amiga: 5 Katalog-Disk. 10,– DM in Briefmarken, MS-DOS Katalog 5,– DM

Selma Fester
A. d. Alpheide 26B, 3070 Nienburg
Telefon 0 50 21/6 49 25

AMIGA PD

Riesiges Angebot ausgesuchter PD Software

- Jedes Programm mit DEUTSCHER ANLEITUNG
- SUPER SICHER – Alles 100 % Virenfrei

zum Sonderpreis PD's ab 1.–

Immer auf dem NEUESTEN · STAND

- Zu jeder AMIGA DOS · AUSGABE die getesteten PD PROGRAMME auf DREI 3,5" Disketten in einem Monatlichen ABO zusammengefaßt für:

DM 19.90 per Bankeinzug DM 22.90 per Nachnahme

Jetzt einmalig für Sie zum Schnupperpreis 9.90

- Fordern Sie unseren Umfangreichen Katalog für nur DM 5,– an



Hotline 0781—57734
von 18.00 - 20.00 Uhr

GERMAN · SOFT
PD – UTE – URLBAUER
Abt. C-64 / C-128 / AMIGA PD
Am Rothweg 9 · D-7600 Offenburg 18
Tel. 0781 / 57734

PUBLIC DOMAIN

Unverschämt preiswert

Disketten sofort lieferbar - Qualitätsdisketten
100 % Umtauschgarantie !

3,5" 2DD/10 St. – 12,50 Tagespreise !

Fuji Disketten	ab 10 St.	ab 100 St.
3,5" 2DD/10 St.	27.-	26.-
3,5" 2DD pink, grün	28.-	27.-

Speichererweiterungen andere auf Anfr.

A 500, 512 KB, abschaltbar	159.-
A 1000, 2 MB Ram Box, 512 KB best.	444.-
A 2000, 8 MB, 2 MB best.	844.-

Drucker:		
	Star LC 24- 10	679.-
	Star LC 10 Color	589.-

andere Modelle auf Anfrage!

Kostenlose Preisliste anfordern, auch für Ausland!

AFM Computer
Postfach 2010, 7886 Murg, Tel. 07763/1234

Amiga Public Domain

24 h Expressversand

ACS	– 229	AMIGAWORLDTOOLCHEST	– 5
ANTARES	– 39	AUGE 4000	– 41
AUSTRIA	– 8	BARRACUDA	– 11
CACTUS	– 35	CHIRON	– 136
DBW RENDER	– 7	DDD	– 13
EROTIC*	– 147	FAUG	– 85
FRED FISH	– 310	FONTS	– 11
FRANZ	– 53	GET IT	– 20b
HIGHLIGHTS	– 33	ICONS	– 8
KICKSTART	– 250	KISS	– 134
NICLAS	– 11	OASE	– 49
PANORAMA	– 32d	P.E.N.I.S.	– 3
PUBLIC-PROJECT	– 5	RAY TRACING	– 7
RPD	– 200	RUHR	– 28
RW	– 17	S.A.F.E	– 36a
TAIFUN	– 120	TBAG	– 32
TORNADO	– 30		

* nur gegen Altersnachweis – Alle Serien ständig aktualisiert

50 Stück 3,5" 2DD Qualitätsdisk **DM 80,–**

Einzeldisk	DM 4,50	***Qualitätsgarantie***
ab 10 Disk	DM 4,00	*Wir kopieren nur mit*
ab 50 Disk	DM 3,50	*doppeltem verify – 100%*
ab 100 Disk	DM 3,00	*virus- u. errorfree. – Etikettiert*

Hardware: 512 KB Speichererweiterung DM 199,–
2 Katalogdisk DM 5,- (V-Scheck/Briefmarken).
Zzgl. DM 4,– bei Vorkasse, bei Nachnahme DM 6,–
Ab 50 Disk bei Vorkasse versandkostenfrei.

ALPHA - SOFT
E. Schlick
Postfach 105 · 6719 Carlsberg
Hotline: 06356-5284

Ihre Hotline

für AMIGA-DOS-TIP

Tel.: 089/4391087 bis -89

Fax: 089/4391080

Mit Aktien Geld verdienen
durch
AMIGA-Börse '90
Das Aktienprogramm für den Profi wie für den Einsteiger

★ Mit den Features vielfach teurerer Programme ★ Zum Beispiel: ★ Konkrete Kaufsignale und Verkaufssignale ★ Aktienzahlen beliebig ★ Charts ★ Gleitende Durchschnitte ★ Kursverfolgung über 4 Jahre ★ Ranglistenwerte sortierbar ★ Technische Aktienanalyse ★ Druckoption ★ Auswahl internat. Aktien auf der Diskette ★ vieles andere mehr ... Dazu eine Benutzeroberfläche vom Feinsten, Pulldown-Menüs, Funktionsfelder, Funktionstastenbelegung, Mausbedienung. Lauffähig auf allen Amigas ab 512 KByte und von der Festplatte!

Programmdiskette und Handbuch DM 178,–
DEMO DM 30,–

zzgl. Versandspesen im Inland inkl. NN DM 12,–, Ausland DM 15,– Wir liefern innerhalb der BRD gegen Nachnahme, in das Ausland gegen Vorkasse (Euroscheck).

GUSSENBAUER
Panoramastr. 18 · 7107 Nordheim
Tel.: 0 71 33/49 25

Speichererweiterungen
ANRUFEN !!!! LOHNT SICH !!!!
TAGESPREISE !!!!!

Amiga 500	- 512 KB intern, abschaltbar	ab DM	159,--
Amiga 500	- 1 MB intern, absch , Uhr	ab DM	398,--
Amiga 500	- 2 MB intern, absch , Uhr	ab DM	598,--
Amiga 1000	- 2 MB extern, absch,	ab DM	798,--
Amiga 2000	- 2 MB intern, absch,	ab DM	898,--

Alle Speichererweiterungen sind autokonfigurierend, abschaltbar und mit sehr schnellen RAM's (100ns und schneller) ausgerüstet! Durch Megabit-Technologie minimaler Strombedarf
****12 Monate Garantie****

Laufwerk 3.5" intern f. Amiga 2000	DM	169,--
Laufwerk 3.5" extern, durchgeschl. Bus, abschaltb.	DM	198,--

B & S Computer-Vertriebs GmbH
Beethovenstr. 33 ; 4172 Straelen 1
Tel: 02834/1249 ; Fax: 02834/6979

MacSoft – AMIGA SHOP
Hardware – Software – Schulung – PD

Public Domain
DISKETTE AUF 2 DD NUR 4,– DM

Über 4500 PD-Disk! Immer aktuell!
24-Std.-Versand-Service
Katalog-Disketten anfordern, 5,– DM
Selber abholen, NN gespart!
Hardware-Zusammenstellung auf Wunsch.
Lassen Sie sich Ihren persönlichen Amiga anfertigen.
Fragen Sie nach unseren Amiga-Einsteiger-Kursen.

Telefon 02 31 / 51 60 10
Mo.-Fr. 10–13, 15–20 Uhr, Sa. 10–16 Uhr
Hannövrischestr. 82, 4600 Dortmund 1
*BTX * mac soft amiga #*

PV COMPUTER SHOP

WIE?... NEU! NEU! NEU! NEU! NEU! AMIGA Schweiz AMIGA Schweiz

WAS?... AMIGA und alles für den AMIGA bis Professional ...

WO?... P.V. COMPUTER-SHOP
Bubikon ZH, Ladengeschäft
Neu! in 8623 Wetzikon Neu!
an der Bahnhofstr. 278
Tel. 01-9307954

HARDWARE • SOFTWARE • BERATUNG

Von 9.30 bis 22.00 UHR

AMIGA PD SO GÜNSTIG WIE NOCH NIEMALS ZUVOR
WIR KOPIEREN MIT VERIFY
ÜBER 4500 PD-DISK

R. Dombrowski
Postfach 71 04 62
2000 Hamburg 71
040/ 642 82 25
NEU 24 Std.Versand-Service ohne Aufpreis.

3,5" 2DD	5,25" 2D
PD incl. Qualitätsdisk	PD incl.Qualitätsdisk
1 - 9 a DM 2,80	10 -39 a DM 1,30
10 -79 a DM 2,00	40 -99 a DM 1, 10
80- a DM 1,90	100- a DM 1,00
Serienabnahme ab 200 PDa 1,70 DM	Serienabnahme ab 300 PD a'0,95 DM
WIR HABEN ›80 SERIEN	ALLE UNSERE SERIEN SIND IMMER AKTUELL ›60 !
ABO Möglichkeit!	EIGENE SERIE "ANTARES"

5,25" Markendisk Zuschlag je Staffel von 0,40 DM a' Disk
5,25" Farbdisketten Zuschlag je Staffel von 0,20 DM a' Disk

PD incL. 3,5" 2DD Markendisk.

1 -9 a DM 3,00	10 -79 a DM 2,40
80- a DM 2,30	SensationelleNeuheiten

ABSOLUT NEU: deutsche Katalogdiskette; auf dieser Diskette ist der Inhalt von 6 normalen Katalogdisketten enthalten incl. ANTARES Menue 4,--DM incl. Portonur bei Vorkasse (Briefm) Nachnahme Vorkasse (nur Scheck oder Überweisung kein Bargeld +Porto: 6,00DM Nachnahme 8,00 DM incl. Verpackung

IMMER UP-TO-DATE AMIGA PD
Wir fuehren (fast)alle Serien
24 Std.Versand

3,5": ab 2,60 DM
5,25": ab 1,60 DM

Verify-Etikettiert-auf Viren geprüft

DTP - PAKET
10 Disketten mit ca.200 Fonts und 200 Clips für Pagesetter,Deluxe Paint usw.Incl.80 Druckertreiber Mit ausführlicher Dokumentation und Illustration

D.Ptak St.Ingbert 06894/381331
A.Fuchs Trier

Musik- und Grafiksoftware Shop
Wasserburger Landstr. 244 * 8000 München 82
Telefon: 089 / 430 62 07

"THE QUEST SEQUENZER"
Das neue 24-Spur Sequenzerprogramm für alle AMIGA.

Das bekannte Sequenzerprogramm 'TEXTURE' ist bereits seit 1985 eines der erfolgreichsten Sequenzerprogramme auf dem IBM. Endlich ist dem Programmautor Roger Powell und Sound Quest die Umsetzung für den Amiga gelungen. TEXTURE wurde durch so bekannte Anwender wie Jan Hammer und Stevie Wonder bekannt.

Die Bedienung erfolgt entweder über die Tastatur oder direkt mit der AMIGA-Maus. Das Programm bedient sich einer ausgefeilten PULL-DOWN-MENÜ-Technik, um eine optimale Bedienerführung zu gewährleisten. Dabei wurde vor allem Wert auf optimales Timing gelegt, sodaß der AMIGA nun auch studiotauglich geworden ist. Zahlreiche Funktionen erleichtern das Aufnehmen, Arrangieren und Manipulieren von Midi-Daten. Alle Funktionen können in Realtime während des Abspielens ohne Timingprobleme aktiviert werden. Der QUEST SEQUENZER läuft auf allen AMIGA Modellen ab 512 KByte RAM und mit allen Standard-MIDI-Interfaces.

Preis: nur DM 298.-

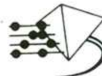
Außerdem führen wir Editoren für viele gängige Synthesizer von Roland, Yamaha, Casio, Ensoniq u.a.

Kostenlosen AMIGA-MIDI-Katalog anfordern! (Rückporto)

Rufen Sie uns einfach an oder besuchen Sie uns in unserem Laden
MO - FR 10 - 13 UHR UND 14 - 18.30 UHR

Daten- und Organisationssysteme
Hard- und Softwarevertrieb

Ihr AMIGA-Fachhändler im Bergischen Land!



Filecard A-2000 AutoBoot

21 MB 40ms MFM	999 DM	47 MB 28 ms RLL	1.499 DM
32 MB 40ms RLL	1.099 DM	88 MB 15 ms MFM	2.849 DM
47 MB 40ms RLL	1.459 DM	133 MB 15ms RLL	3.399 DM

Alle Cards inkl. ALF 2.0 Software, AutoPark, AutoBoot unter Kick 1.3, Einbau im Preis.

HardDisk A-500/1000 AutoBoot

20 MB 40ms MFM	1.149 DM	48 MB 28ms RLL	1.649 DM
32 MB 40ms RLL	1.239 DM	65 MB 28ms RLL	1.799 DM
43 MB 28ms MFM	1.599 DM		

Alle Disks inkl. ALF 2.0 Software, Gehäuse, Netzteil
Amiga Expansionsport ist durchgeführt, Anschlußkabel

Preisgünstige Speichererweiterungen

abakus 512KB, m. Akku, abschaltbar	199 DM
A 508 2MB-Card, Akku, abschaltb., 512KB bestückt	349 DM
A 508 2MB-Card, Akku, abschaltb., 2MB bestückt	649 DM
2-MB-Box, Akku, ab- u. umschaltb.,512KB bestückt	449 DM
4MB-Box, Akku, ab- u. umschaltb., 2 MB bestückt	999 DM
MICROBOTICS 8-UP 2,4,6,8MB Betrieb, 2 MB bestückt	899 DM
MICROBOTICS 8-UP 8 MB bestückt nur	1.999 DM

Kickstart-Umschaltplatinen

Kick-Umschaltplatine ROM/ROM mit ROM 1.2	69,90 DM
Kick-Umschaltplatine ROM/ROM mit ROM 1.3	99,90 DM
Kick-Umschaltplatine ROM/EPROM	69,90 DM
BootStrap, mit RAM, mit WOM, mit Gary Adap	339 DM

Öffnungszeiten (Büro + Ladengeschäft)
Mo – Fr 10.00 – 18.30 – Sa 10.00 – 14.00 – langer Sa 10.00 – 16.00
Sedanstraße 136 * 5600 Wuppertal 2
Tel. 0202/50501500 * Martin Kramer

DONAU-SOFT
24 h-Schnellversand

Neutrale Disketten
3,5" 2DD (100 % errorfree)

	von Sentinel	von SONY/Collossus
bis 99 Stück	1,60 DM	2,00 DM
ab 100 Stück	1,40 DM	1,85 DM
ab 500 Stück	**1,25 DM**	**1,70 DM**

Laufwerke mit allen Extras

3,5" intern	155,– DM
3,5" extern, abschaltbar, Busdurchführung	209,– DM
5,25" extern, wie 3,5" + 40/80-Trackumschaltung	269,– DM

Sim City	89,– DM	GFA-Basic	208,– DM
B.A.D.	77,– DM	GFA-Compiler	129,– DM
DPaint III dt	240,– DM	512 KB-Erw. (A500)	208,– DM
Zoetrope 1.1	189,– DM	2 MB-Erw. (A500)	598,– DM

Vorkasse: + 5,– DM, Nachnahme: + 8,– DM, Ausland: + 10,– DM

MAIK HAUER
Postfach 14 01, 8858 Neuburg
Tel.: 0 84 31/4 97 98 (bis 22 Uhr)
Fax: 0 84 31/4 98 00
BTX: *Donau-Soft #

Die PD-Werkzeugkiste

Nachdem wir uns in der letzten Ausgabe etwas mehr den Gag-Programmen gewidmet haben, wollen wir heute wieder einmal ein paar kleine Helferlein vorstellen, die den Umgang mit dem Amiga leichter gestalten sollen.

Die Franz-Diskette Nr. 01 umfaßt sowohl kleine Spielchen als auch nützliche Tools. Außerdem bietet sie dem Programmierer verschiedene Programme, deren Sourcecodes ebenfalls auf der Diskette enthalten sind.

Das erste Programm heißt **Dazzle**, das sich auf einer der ersten Fish-Disketten befand. Es erzeugt mit Hilfe einer mirror (Spiegel) -ähnlichen Funktion sehenswerte Zufallsgrafiken, die sich ständig verändern. Der zugehörige C-Source ist auf der Diskette enthalten.

Bei **GFXMEM** handelt es sich um eine nützliche Speicheranzeige, die in C geschrieben wurde. Es wird in einem separaten Fenster der freie Speicher in Form von zwei Balkengrafiken dargestellt, die das Chip- und das Fast-RAM symbolisieren. Auch bei diesem Programm wurde der Source nicht vergessen.

Liebe Leser,

Die Programme, auf die wir im Rahmen der PD-Seiten eingehen, befinden sich größtenteils auch auf anderen PD-Reihen. Die Preise für die Disketten variieren jedoch von Vertreiber zu Vertreiber. Wenn Sie sich für ein bestimmtes Programm interessieren, sollten Sie in eigenem Interesse einen Preisvergleich anstellen.

Clock beinhaltet ebenfalls eine Anzeige über den freien Speicher. Außerdem stellt es zusammen mit der Speicheranzeige auch die aktuelle Uhrzeit in einem Fenster auf dem Titelbalken dar. Das nächste Programm auf dieser Franz-Diskette ist **KickChange**. Es bietet den Besitzern eines Amiga 1000 die Möglichkeit, die Kickstartversion zu wechseln, ohne jedesmal den Computer ausschalten zu müssen. Wer die Funktion dieses Programmes näher untersuchen möchte, findet den BASIC-Sourcecode dazu ebenfalls auf der Diskette.

Für C-Programmierer ist **Palette** interessant. Auch hier liegt der Sourcecode des in C geschriebenen Programmes auf der Diskette vor. Es dient dazu, die mit dem Programm Preferences eingestellten Farbwerte der Workbench nachträglich zu ändern. Die Farben lassen sich durch einen Schieberegler einstellen und werden als Hexadezimalzahlen dargestellt.

FontEdit ist, wie der Name schon vermuten läßt, ein Zeichensatz-Editor. Er erlaubt es, beliebige Fonts (Zeichensätze) von Diskette einzuladen und nachzubearbeiten oder ganz einfach neue zu erstellen. Sowohl die Erstellung der Zeichen als auch die Bedienung erfolgt komplett mit der Maus. Das Programm bietet alle notwendigen Optionen, beispielsweise das Verändern der Baseline oder Breite und Höhe der Zeichen. Außerdem ist es möglich, proportionale Zeichensätze zu erstellen.

Nun aber zu einem negativen Punkt des Programms. Es kam beim Austesten des Zeichensatz-Editors mit einem AMIGA 2000 und KICKSTART 1.3 zu gelegentlichen Abstürzen. So findet man auch in der Anleitung den Vermerk, daß das Programm nur mit einem AMIGA 1000 und KICKSTART 1.1 ausprobiert wurde und nur mit dieser Anlage ein reibungsloser Betrieb garantiert werden könnte. Nichtsdestotrotz kann man mit dem Programm aber über weite Strecken gut arbeiten.

Das Tool **MenuEditor** verspricht allen Besitzern der Betriebssystemversion KICKSTART 1.1 das einfache Erstellen eigener Menüs und das Einbinden in selbsterstellte C-Programme.

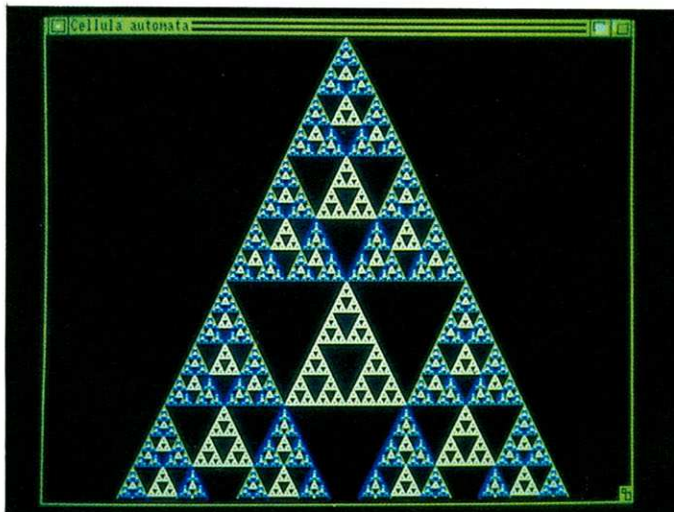
MyCLI stellt eine Ergänzung zum Original-CLI dar. Zum Beispiel erlaubt es die Belegung der Funktionstasten mit beliebigen Texten. Um die Taste F1 mit dem Text "dir df1:c/" zu belegen, schreibt man einfach den Befehl: DEF F1 dir df1:c/ . Außerdem stellt MyCLI viele Befehle zur Verfügung. Die wichtigsten Kommandos stehen mit DEF, HELP, OFFLINE und TERMINAL ins Haus – pardon, in den Amiga. Mit dem letzten Befehl gelangt der Benutzer in einen Terminal-Modus, der es gestattet, direkt mit einem VAX-Rechner in Verbindung zu treten. Alle Tasteneingaben werden sowohl auf dem Bildschirm, als auch an der seriellen Schnittstelle ausgegeben. Der Befehl ENDCLI funktioniert mit KICKSTART 1.2 (oder höher) nicht immer einwandfrei. Ansonsten stellt MyCLI eine nützliche Hilfe dar.

Als nächstes Programm findet der PD-Interessierte **SuperBitMap**. Dabei handelt es sich um eine Demonstration der grafischen Fähigkeiten des Amiga. Es werden Linien auf eine übergroße Bitmap gezeichnet, und dabei ist es möglich, mit den Tasten u,d,l,r die Grafik in alle Richtungen scrollen zu lassen. Das Programm bietet etwas fürs Auge und für Programmierer. Wer der Hochsprache C mächtig ist oder, besser noch, nicht ganz mächtig ist, der findet auf der Franz-Diskette Nr. 01 das C-Sourcelisting dieses Programms.

Das letzte Programm auf der Franz-Disk nennt sich **SpeechToy**. Bei diesem Programm handelt es sich um ein Sprachausgabeprogramm, bei dem sich alle Einstellungsmöglichkeiten über Gadgets und Schieberegler kontrollieren lassen. Als Besonderheit stellt SpeechToy die Möglichkeit zur Verfügung, ein Gesicht anzuzeigen, das beim Sprechen den Mund bewegt. Das Programm kann komplett mit der Maus gesteuert werden.

(Ingmar Reyer/br)

Name: Franz Nr.01
Quelle: APS-Electronic



Das Programm Cell-Auto auf der Diskette Franz Nr.47 – so schön kann Geometrie sein

Die Franz-Diskette Nr.47 ist besonders für Programmierer geeignet, die sich etwas in Assembler auskennen. Aber auch die Computerbenutzer, die sich für Mathematik interessieren, kommen nicht zu kurz.

Beispielprogramme für den A68k-Assembler

Das Programm **Intro_Maker** bietet Spieleprogrammierern die Möglichkeit, einen Vorspann mit soft-scrollendem Sternenhintergrund und Text zu erstellen. Der Text, der von rechts nach links bewegt wird, kann bis zu 5000 Zeichen enthalten. Mit diesem Tool wird dem Benutzer viel Zeit erspart, die man nun mehr für das Hauptprogramm zur Verfügung hat. Die Scrollmessage kann dann unter einem beliebigen Namen abgespeichert werden und ist dann eigenständig lauffähig.

Im Verzeichnis A68K_Beispiele findet der Assembler-

Fan einige Programme mit Quellcode. Das erste File heißt **Cell-Auto**. Es zeichnet sogenannte Sierpinsky-Tetraeder, also Grafiken, die sich aus Dreiecken aufbauen. Die Außenform ist ein großes Dreieck, das mit vielen kleineren gefüllt ist. Das Programm ermöglicht es, die Grafik mit fünf Parametern zu verändern.

Lissajous zeichnet gleichnamige Figuren. Bei diesem Programm ist besonders der Quellcode interessant, da es den Umgang mit der Mathtrans.library recht gut veranschaulicht.

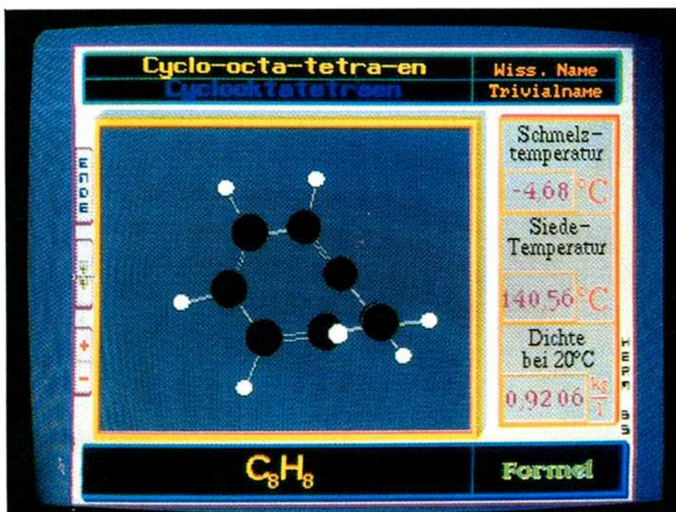
Weiterhin sehr interessant ist das Programm **Say**. Es liest dem gestreßten Benutzer ASCII-Dateien vor. Dabei wird immer eine Zeile separat auf dem Bildschirm angezeigt und vorgelesen.

Für Mathematiker ist wahrscheinlich **Prime** von Nutzen. Es gibt in rasender Geschwindigkeit Primzahlen in einem Fenster aus.

Plane hilft dem Programmierer beim Verständnis der graphics.library. Der Quellcode demonstriert das Linienzeichnen in Perfektion. Startet man das Programm, so wird ein Flugzeug gezeichnet, wieder gelöscht, dann um einen bestimmten Winkel gedreht und wiederum gezeichnet. Dieser Vorgang wiederholt sich ständig, bis das Programm abgebrochen wird.

Das Programm **Dump** ist ein äußerst nützliches Tool: es handelt sich hierbei um einen Speichermonitor, der den RAM-Speicher des Amiga als ASCII-Zeichen, als Hex-Zahlen oder in disassemblierter Form darstellt. So ist es möglich, einen tieferen Blick in das Betriebssystem oder auch in diverse Programme zu werfen. Die Bedienung stellt kein Problem dar. Die gesamte Eingabe läßt sich benutzerfreundlich mit der Maus steuern. Weiterhin bietet Dump viele Optionen. So kann der User nach bestimmten Zeichenfolgen suchen oder mit dem Menüpunkt "Micro" den verwendeten Mikroprozessor auswählen (68000, 68010 bzw. 68020).

Boing ist wieder etwas für Spieleprogrammierer. Der Quellcode wird dieser Gruppe helfen, einige Geheimnisse der Spriteprogrammierung zu lüften. Boing stellt zwei Sprites dar, die sich schnell über den Workbench-Screen bewegen.



Molekülstrukturen werden auf der Franz Nr.397 dreidimensional dargestellt. Die Moleküle können dabei auch animiert werden

Das letzte Demoprogramm für den A68K-Assembler nennt sich **Cmp**. Cmp ist aber nicht nur ein einfaches Demo. Nein, es ermöglicht das Vergleichen zweier Programme (ähnlich dem diff-Programm) und zeigt deren Unterschiede an. Bei der täglichen Arbeit am Computer ist dies natürlich sehr hilfreich. Hat man beispielsweise den Verdacht, einen Linkvirus auf seiner Diskette als Gast zu haben, benutzt man einfach Cmp und vergleicht alle Dateien mit denen auf einer (hoffentlich) angelegten Sicherheitscopy. Wird ein Virus geortet, kann man wirksam gegen ihn vorgehen.

Corral ist ein Abkömmling der vielen Fraktal-Generatoren oder Apfelmännchenprogramme. Mit ähnlichen Formeln zeichnet es aber keine "dickbäuchigen Männer", sondern bizarre Korallenstrukturen. Die erstellten Grafiken sind wirklich sehenswert, aber "dank" der komplexen Formeln auch nicht gerade sehr schnell berechnet.

(Ingmar Reyer/br)

Name: Franz Nr.47
Quelle: APS-Electronic

Eine fulminante Lücke im Softwarebereich besteht immer noch bei den Chemieprogrammen. Zwar wird mit solchen Programmen nur eine kleine Interessengruppe angesprochen, aber auch Außenseiter sollten auf ihre Kosten kommen. Dies sagte sich bestimmt auch Holger Franz, als er die Diskette Nummer 39 seiner Franz-Serie zusammenstellte.

So enthält diese Diskette ausschließlich ein Chemieprogramm namens **Organische Moleküle**. Nach dem Booten wird dem Benutzer die Möglichkeit geboten, eine RAM-Erweiterung zu nutzen. Mit dieser Erweiterung wird das ständige Nachladen der Moleküle vermieden. Vorab öffnet sich ein Titelbild mit der Aufschrift 'Organische Moleküle in Rotation'. Dem Titel nach zu urteilen, beschäftigt sich das Programm also mit der Darstellung dreidimensionaler Moleküle im Raum. Über den rechten Bildschirmrand erstreckt sich senkrecht ein Zahlenmenü, das von eins bis neun reicht. Hinter diesen Zahlen verbergen sich die einzelnen Moleküle, die in dem Programm vorkommen.

Molekülstrukturen genauer betrachtet

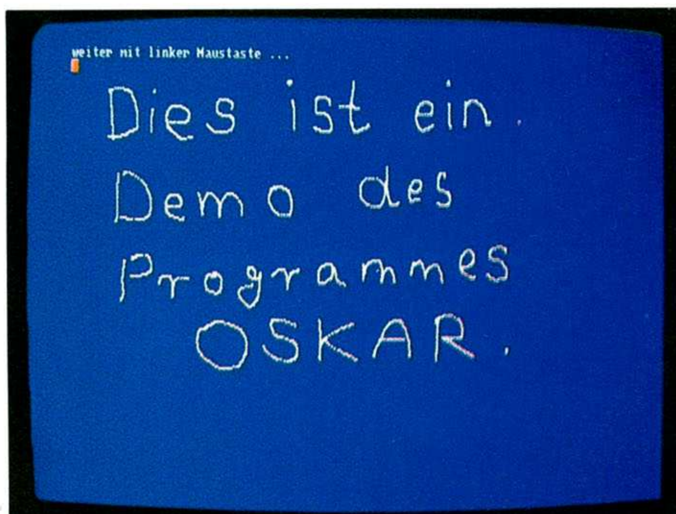
Nach dem Anklicken einer Zahl wird das betreffende Molekül eingeladen und auf dem Bildschirm dargestellt. Dem Anwender bietet sich nun eine Benutzeroberfläche, in der er Näheres über das ausgewählte Molekül erfahren kann.

Inmitten des Bildschirms wird nach dem Anklicken eines Start-Requesters das betreffende Molekül räumlich animiert. Wie in der Chemie üblich, werden bei der Darstellung die Atome dabei mit verschieden Farben und Größen gekennzeichnet:

Kohlenstoff	=	schwarz
Wasserstoff	=	weiß
Sauerstoff	=	rot

Die Bindungen zwischen den einzelnen Atomen sind mit Strichen gekennzeichnet, so daß auch Doppel- bzw. Dreifachbindungen bei dieser Darstellung sofort erkannt werden können.

Die Geschwindigkeit, mit der sich das Molekül dreht, kann durch zwei Buttons verändert werden. Am unteren Rand des Bildschirms zeigt sich dem Benutzer die Strukturformel des Moleküls. Die dazugehörige Dichte, Schmelz- und Siedetemperatur findet man an der rechten Seite. Mit der Nennung des Trivial- und wissenschaftlichen Namens ist das Programm schließlich voll ausgekostet. Schade ist jedoch, daß man keinen Einfluß auf die Moleküle hat oder sich gar selbst welche erstellen kann.



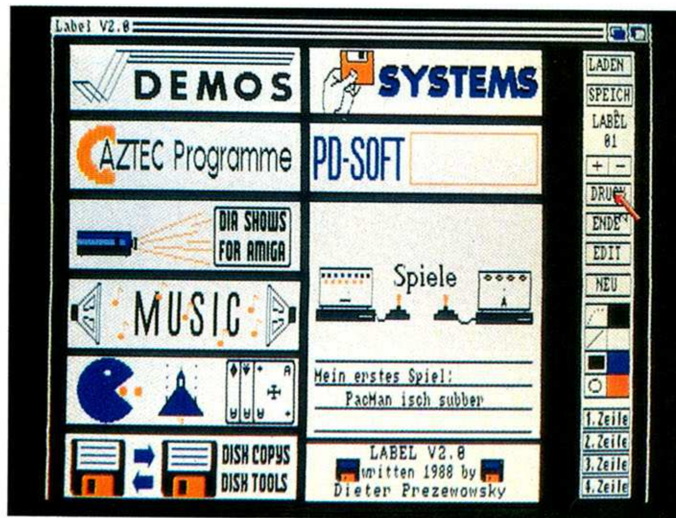
Mit dem Programm Oskar von der Bavariansoft Nr.91 lassen sich nette Vorspanne erzeugen

Es sind lediglich neun Moleküle vorgegeben. Hier eine Liste der im Programm enthaltenen Moleküle:

- Cyclooktatetraen (C_8H_8)
- Isopropanol(CH_3-CHOH-CH_3)
- tert.Butanol (CH_3-C(CH_3)OH-CH_3)
- Cyclohexan (C_6H_{12})
- Vinylethin (C_4H_4)
- Dichlorbenzol ($C_6H_4Cl_2$)
- Acetonphenon (CH_3-CO-C_6H_5)
- Styrol (C_6H_5-CH – CH_2)
- Trimethylamin $(CH_3)_3N$

(Jürgen Seibel/br)

Name: Franz Nr.397
Quelle: APS Electronic



Mit dem Etikettendruck-Tool Label V2.0 von der Bavariansoft Nr.40 lassen sich tolle Diskettenaufkleber erstellen

Es ist immer wieder überraschend, was den Amigabenutzern auf PD-Disketten angeboten wird. Das Thema der Bavariansoft Nr.40 ist das Ausdrucken von Etiketten. Dazu werden dem Benutzer sieben Etikettenprogramme zur Auswahl gestellt.

Labels der Luxusklasse

Das Programm **Label V2.0** bietet ein hohes Maß an Funktionen. So stehen dem Benutzer anfangs acht verschiedene Diskettenlabel-Grafiken zur Verfügung. Auf der Diskette befinden sich jedoch weitere fünfzehn Bilder, die durch eine Einlade-Option angesprochen werden können.
Am rechten Bildschirmrand befinden sich die Optionen zum Erstellen eines Etiketts. Dabei spielt die Edit-Funktion die entscheidende Rolle. Nach dem Anklicken dieses Menüs können Sie eigene Label-Grafiken erstellen. Hierzu steht Ihnen, wie bei DPaint ein Malmodus zur Verfügung. Per Maus werden die Koordinaten für ein Rechteck, einen Kreis oder lediglich eine Linie definiert. Die Texteingabe ist unabhängig vom Edit-Mode. Nur wenn dieser Modus deaktiviert ist, kann mit der Eingabe begonnen werden. Als Druckertreiber dienen die Preferences, so daß eine Druckeranpassung keine Probleme aufwerfen sollte.
Das nächste Etikettenprogramm namens **Etiketten V1.02** birgt nichts Aufregendes. Das Programm wird über drei auf dem Bildschirm erscheinende Schalter gesteuert. Diese Schalter dienen zur Eingabe, zum Drucken und zum Beenden des Programms. Ein Manko dieses Programms ist jedoch, daß die Daten nicht gespeichert werden können.
Der **Disk-Label-Drucker** ist komplett in AmigaBASIC geschrieben. Je nach Speicherplatz gliedert sich das Programm. Im Falle einer Speichererweiterung können zusätzliche Optionen genutzt werden. Hierbei handelt es sich um zwei Programmfiles, die zum Hauptprogramm geladen werden. Sie ermöglichen das Umgestalten von Labelgrafiken. Dazu wird ein IFF-Bild mit einem Malprogramm (beispielsweise DPaint) bearbeitet. Nach dem Einladen des Hauptprogramms prüft der Rechner, welche Laufwerke angeschlossen sind. Danach erscheint ein Titelbild, das die Benutzeroberfläche darstellt. Durch das Anklicken diverser Menüs erreicht man die Texteingabe. Mit dem Anklicken des Bildmenüs öffnet sich ein Fenster, in dem sich die zur Auswahl stehenden Grafiken befinden. Die momentan ausgewählte Grafik ist umrahmt.
Das Programm **Etti** (nicht zu verwechseln mit unserem Listing aus Ausgabe 3/90) ist ebenfalls in AmigaBASIC geschrieben, jedoch enthält dieses Programm weit mehr Optionen als das eben beschriebene Etiketten V1.02. Durch Tastaturkommandos können die folgenden Optionen ausgeführt werden:
Drucken, Color, Speichern, Laden, Verzeichnis, Reset, Quit, Text eingeben, Farbe ändern, Art der Schrift, Unterstreichen, Up and Down.
Das Verzeichnis Option dient zum Erstellen eines neuen Directories, dessen Name frei wählbar ist. Weitaus interessanter ist jedoch der Color-Befehl. Aufgrund dieses Aufrufes können Sie endlich die Kapazitäten eines (eventuell vorhandenen) Farbdruckers nutzen. So kann man die Zeilen in den Farben schwarz, rot, blau, violett, gelb, lila und grün erscheinen und ausdrucken lassen. Ein wenig Pep wird nicht allein mit Farben eingeführt, sondern auch durch das Wählen der verschiedenen Schriftarten.
Mit **Etikettendruck** reiht sich ein weiteres Programm in diese Gruppe ein. Das Programm wird über ein Pulldown-Menü bedient, das über vier Hauptmenüs verfügt (Optionen, Druckfunktionen, Run und Windows). Im ersten Menü werden das Directory und die Unterverzeichnisse angezeigt. Zusätzlich kann das Programm in diesem Menü beendet werden. Das Druckfunktionen-Menü enthält einige nicht zu verachtende Optionen. Beispielsweise läßt sich hier der Drucker justieren. Dazu wird der Drucker in Bereitschaft gesetzt, worauf dann ein Probedruck zum Drucker geschickt wird. Des weiteren übergibt man in diesem Menü die Parameter zum Ausdruck. Hier folgt nach der Eingabe des linken Randes die Abfrage der Zeilenanzahl. Nach der Definition der Schriftart ist die Eingabe beendet. Auch bei diesem Programm besteht die Möglichkeit, die Schriftarten für jede Zeile zu variieren. Der Window-Befehl läßt das gesamte Listing des Programms auf dem Bildschirm erscheinen, so daß einzelne Routinen des Source-Codes nach Wunsch umgeschrieben werden können.

Von **Labelprint** existieren zwei Versionen: Labelprint und LabelprintE, die jedoch in ihrer Funktionsweise völlig identisch sind. Ihr einziger Unterschied liegt bei den Druckertreibern. LabelprintE ist für alle Epson-kompatiblen Drucker geschrieben. Dagegen spricht Labelprint den Drucker mit dem PCprtCommand-Befehl an. Bei LabelPrint schießt einem geradezu die Anzahl der Laufwerke, die unterstützt werden können, in die Augen. Man kann in diesem Programm auf vier Laufwerke, die RAM-Disk und auf eine Harddisk zurückgreifen. Ein residenter Search-Befehl sortiert die Fileliste vor dem Ausdruck ständig. Mit **Eti** präsentiert sich das letzte auf der Diskette befindliche Programm, das sich nur aus dem CLI starten



Auch das Programm BootIntro ist ein nettes Utility, mit dem man Scrolltexte automatisch beim Booten ausgeben kann

läßt. Eti ist ein Etikettendruckprogramm, das zwar spartanisch aufgebaut, aber dennoch alle wichtigen Fähigkeiten eines Label-Druckprogrammes beinhaltet. So öffnet sich nur ein Fenster, in das der Text eingegeben wird. Im Projekt Menü kann man die Labels dann abspeichern und in Schönschrift ausdrucken lassen.

(Jürgen Seibel/br)

Name: Bavariansoft Nr.40
Quelle: Bavariansoft



Guru-Meditations simulieren – mit dem Guru-Maker ist dies kein Problem

Bootblöcke werden gerne dazu genutzt, Messages beim "Hochfahren" auf den Bildschirm zu bringen. Dies hat einen großen Vorteil gegenüber normalbootenden Disketten, da im Falle eines Bootblock-Virusbefalles die erwartete Message nicht mehr erscheint. Auf der Bavariansoft Nr. 91 sind nun einige Utilities, die das Erstellen von Bootblöcken und Programmvorspannen zum Kinderspiel werden lassen.

Bootblöcke und Programmvorspanne im Eigenbau

Sicher kennen Sie Oskar, den schnellsten Zeichner aus dem Fernsehen. Aber nicht nur der Fernsehzeichner kann schnell zeichnen – nein, auch der Amiga kann's. Das Programm **Oskar** (wie der Programmierer nur auf diesen Namen gekommen ist...) ist ein kleines Vorspannprogramm, das eine mit der Maus erstellte Grafik (oder Text) auf dem Bildschirm ausgibt. Das Programm wird aus dem CLI mit dem Befehl OSKAR *Dateiname* aufgerufen, worauf man mit der Maus einen beliebigen Text (oder auch eine Grafik) schreiben bzw. malen kann. Das Programm erzeugt nun eine Datei, die mit dem CLI-Befehl TRICK *Dateiname* DELAY das Gezeichnete wiedergibt. Mit dem Zusatz DELAY hinter dem Dateinamen wird das Maß der Verzögerung angegeben, mit der die Datei auf den Bildschirm gebracht wird.

Bei dem Programm **Marquee** handelt es sich um einen Titelgenerator für Laufschriften. Man erzeugt mit einem beliebigen Texteditor (beispielsweise dem ed des AmigaDOS) eine ASCII-Datei und speichert sie unter einem beliebigen Namen ab. Nach Aufruf der Datei MakeText muß zunächst der komplette Pfad zum Marquee-File und dann die Textdatei angegeben werden. Ist dies geschehen, so öffnet sich nach kurzer Zeit ein Bildschirm, in dem der Text als Laufschrift gescrollt wird, dieser wird zudem mit einem Musikstück (der Entertainer aus dem Film 'Der Clou') untermalt.

Mit dem **Bootwriter** wird ein Programm im Bootblock einer Diskette abgelegt, bei dem sowohl der Hintergrund als auch die Laufschrift gescrollt wird. Die Laufschrift kann dabei frei editiert werden. Der Programmaufruf erfolgt aus dem CLI, worauf der Benutzer aufgefordert wird, den Hintergrund festzulegen. Hierbei stehen dem Anwender verschiedene Hintergrundgrafiken zur Verfügung. Nun braucht nur noch der Lauftext eingegeben und abgespeichert zu werden und fertig ist der neue Bootblock.

BootIntro ist ein Bootblock-Utility, das wir schon in der Ausgabe 2/90 vorgestellt hatten. Auf einem farbigen Hintergrund wird eine feststehende Kopfzeile und ein horizontal scrollender Fließtext dargestellt. Die Überschrift kann maximal 20 Zeichen und die Laufschrift maximal 225 Zeichen umfassen.

Der nächste Display-Hack nennt sich **Startle** und basiert auf dem Programm Starts von Leo Schwab. Zuerst erstelle man sich bei diesem Programm eine beliebige Textdatei und speichere sie ab. Der Aufruf von Startle erfolgt wiederum aus dem CLI mit der Syntax *startle* <Textdatei>, wobei Parameter für den Zeichensatz und die Zeichenfarbe ergänzt werden können. Der Text scrollt von unten nach oben über den Bildschirm, während der Benutzer den Anschein hat, er würde durch den Weltraum fliegen.

Guru-Maker ist das letzte Programm auf der Bavariansoft Nr.91. Mit diesem Tool können Sie die Ausnahmezustände (besser bekannt als Guru-Meditations) simulieren. Zwei Eingabezeilen stehen dabei für den Text zur Verfügung, der maximal 27 Zeichen lang sein darf. Durch Aktivieren eines Test-Buttons kann sich der Benutzer sein Werk vorab ansehen. Laut Autor kann man sich diese Message auf dem Bootblock abspeichern, die dann beim Booten der Diskette auf dem Bildschirm erscheint. Diese Funktion versagte jedoch bei unserer Version den Dienst.

(br)

Name: Bavariansoft Nr.91
Quelle: Bavariansoft

Eine kleine Übersicht der Vertreiber von Public Domain, Free- und Shareware sowie Prüf-vor-Kauf-Programmen (ohne Anspruch auf Vollständigkeit)

A.P.S. Electronic
Sonnenborstel 31
3071 Steimbke
Tel.: 05026/1700

BAVARIANSOFT
Postfach 72
8473 Pfreimd
Tel.: 09606/286

BELLINGRATH, CHRISTIAN
Hans-Böckler-Str. 55
5860 Iserlohn
Tel.: 02371/24192

DIGITAL MARKETING D. MÜCKTER
Krefelder Str. 16
5142 Hückelhoven-Baal
Tel.: 02435/2086,428 oder 1295

HIESKE, DIETER
Schillerstr. 36
6700 Ludwigshafen
Tel.: 0621/673105

KEIM, PETER
Vogelsanger Str. 34
5000 Köln 30
Tel.: 0221/520765

MAXON COMPUTER GMBH
Industriestr. 26
6236 Eschborn
Tel.: 06196/481811

Amiga PD-Depot
Wolfgang Bittner
Keltenstr. 15
6700 Ludwigshafen
Tel.: 0621/674974

TechnikSupport Verlag GmbH
Bundesallee 36-37
1000 Berlin 31
Tel.: 030/8621314

WOLF Computertechnik
Deipe Stegge 187
4420 Coesfeld
Tel.: 02541/2874

Österreich

KÜPPERS, BERND
Felberstr. 7
A-5730 Mittersill
Tel.: 06562/282

PETER RAUSCHERS COMPUTERSHOP
A-1100 Wien
Weldengasse 41
Tel.: 0222/621535

Public-Domain-Spieleshow

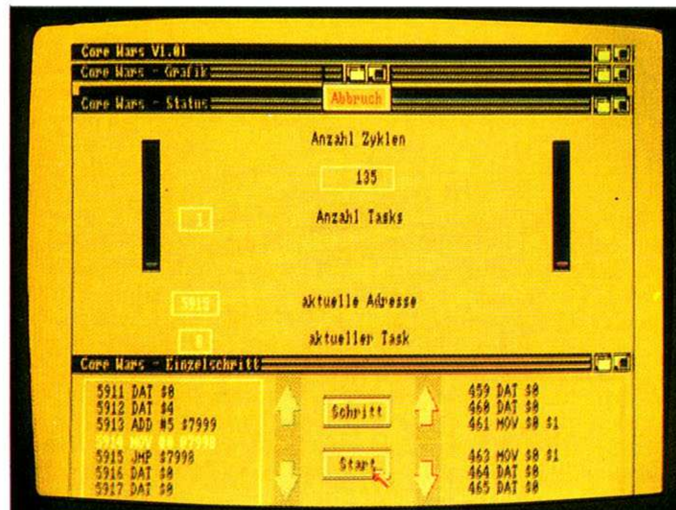
Nicht immer haben Public-Domain-Spiele kommerzielle Vorbilder. Andere Produkte orientieren sich an Brettspielen – Themen, die für die Softwareindustrie erst bei Bestsellern wie Trivial Pursuit interessant werden. Public-Domain-Programme entstehen zumeist aus Spaß an der Freude. So gibt es für Public-Domain-Anwender viele kleine Leckerbissen, mit denen der kommerzielle Softwaremarkt nicht aufwarten kann.

In den USA ist das Spiel Core Wars auf PCs und Macintoshs schon weit verbreitet, in Europa ist es jedoch noch so gut wie unbekannt. Dies soll mit der Amiga-Version als Public-Domain-Software anders werden.
Bei dem Spiel handelt es sich um ein Programm, das "Kampfprogramme" gegeneinander antreten läßt und diesen Vorgang grafisch darstellt. Der Begriff "Kampfprogramm" bedarf wahrscheinlich einer Erklärung: Diese Programme werden in einer verkürzten Assemblersprache geschrieben.

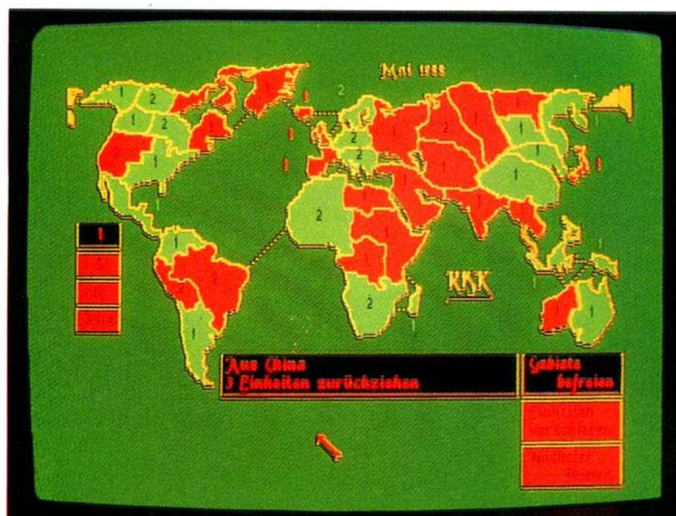
Core Wars

Bei Core Wars nennt sich die Assemblersprache Redcode und umfaßt zehn Befehle. Mit Hilfe dieser Befehle ist es möglich, Speicheradressen zu bombardieren, das heißt, bestimmte Adressen werden mit Nullen überschrieben. Außerdem kann man sein eigenes Programm im Speicher verschieben oder reparieren. Wenn man mit diesen Möglichkeiten geschickt umgeht, kommen dabei erstaunliche Programme heraus. Sie haben zum Ziel, ein anderes, ebenfalls im Speicher stehendes Programm zu zerstören. Sie werden sich jetzt fragen: Was haben denn dabei die Mitspieler zu tun? Die Antwort ist einfach. Der Computer spielt gegen sich selbst das von den Programmierern (Spielern) erstellte Programm. Er arbeitet dabei zwei Kampfprogramme abwechselnd und schrittweise ab. Das interessante daran ist, zu sehen, wie das eigene Programm beim Kampf abschneidet. Es ist schon toll, wenn man sieht, daß das in stundenlanger Arbeit erstellte Programm das des Gegners im Kampf besiegt.
Im Programmieren Ungeübte brauchen jetzt aber nicht den Kopf hängen zu lassen. Es stellt selbst Anfänger vor keine großen Probleme, mit Hilfe der sehr ausführlichen und guten Beschreibung (80 Seiten) kleinere Programme schon nach einer halben Stunde selbst zu erstellen. Core Wars enthält einen schnellen Editor, mit dem man sehr komfortabel programmieren kann, einen Assembler, der die Befehlsfolgen in für den Computer verständliche Befehle übersetzt, und ein Ablaufprogramm, mit dem es möglich ist, zwei Kampfprogramme gegeneinander antreten zu lassen. Das Ablaufprogramm stellt den Speicher (das Spielfeld) grafisch dar. Damit ist leicht nachzuvollziehen, wie die Programme arbeiten.

Name: Core Wars
Quelle: Panorama 30b, Franz 21



Krieg der Programme – bei Core Wars wird der Speicher Ihres Amigas zum Schlachtfeld



Eine Amiga-Umsetzung des beliebten Risiko-Brettspiels

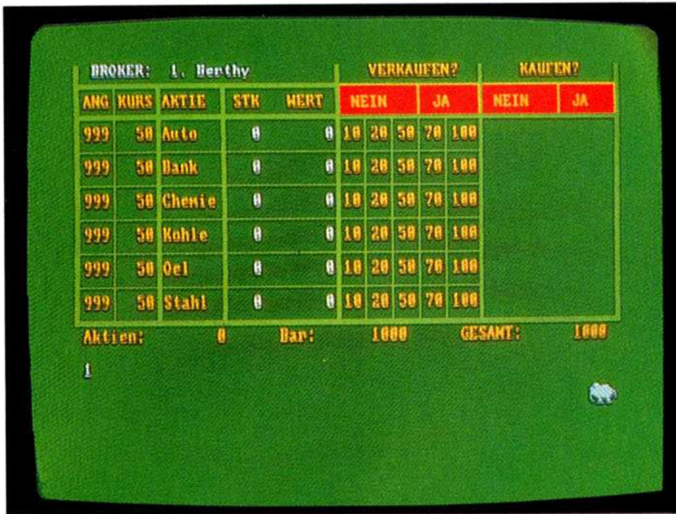
Ein, nach wie vor sehr beliebtes Brettspiel ist Risiko. Nun gibt es eine Public-Domain-Version für den Amiga, die sehr gut gelungen ist. Das Spielfeld von Risk besteht aus einer Weltkarte mit sechs Kontinenten und 42 Ländern, die auf bis zu fünf Mitspieler aufgeteilt werden.

Risk

Von diesen fünf Mitspielern kann der Computer beliebig viele simulieren. Pro Spielrunde erhält jeder Spieler zusätzliche Truppen, um möglichst viele Länder besetzt zu halten oder erobern zu können. Je mehr Länder von einem Spieler besetzt sind, desto größer ist die Anzahl neuer Truppen, die er pro Zug hinzubekommt. Dies ist auch bitter nötig, da die Truppenstärke bei jedem Durchgang stark dezimiert wird. Das Spiel hat der gewonnen, der es schafft, alle Länder zu besetzen oder eine zu Beginn des Spieles gestellte Aufgabe zu lösen. Bei den Kämpfen, die dabei unerläßlich sind, spielt der Zufall in Form von Würfeln mit. Je mehr Truppen ein Land angreifen oder es besetzen, desto mehr Würfel hat der Spieler. Allerdings sind drei Würfel das Maximum. Der, der die größte Augenzahl erreicht, bezwingt eine Truppeneinheit des Gegners. Das Würfeln wird so oft wiederholt, bis der Angreifer oder der Besitzer des Landes keine Truppen mehr im umkämpften Gebiet hat.
Risk sorgt mit seinem taktischen Element und der Fülle einstellbarer Optionen für lange anhaltenden Spielspaß. Die Mausbedienung des Programms ist problemlos, und die Grafik ist schön gezeichnet. Ein Minuspunkt muß dennoch angemerkt werden. Es ist mit 512 kByte Speicher nicht möglich, das Programm von der Workbench zu starten.

Name: RISK
Quelle: Oase

Würden Sie gern das Handeln mit Aktien trainieren? Sie glauben, Ihnen fehlt das nötige Geld dazu? Mit dem Simulationsprogramm Broker ist es möglich, ohne eigenes Kapital das richtige Spekulieren mit Aktien zu üben.
Broker bietet den bis zu vier Mitspielern die Möglichkeit, Aktien zu erstehen und abzu-



Aktien und Wertpapiere – darum dreht sich an der Börse und in dem PD-Spiel Broker alles



Gedächtnistraining spielerisch – es gilt, Ton und Farbfolgen fehlerfrei zu reproduzieren

stoßen. Dabei müssen sie Steuern, Optionen, Dividenden und Kontozinsen berücksichtigen. Weiterhin kann jeder Spieler einen Kredit aufnehmen, den er plus Zinsen im Verlauf des Spieles abzahlen muß. Die Kurse der Aktien berechnen sich aus den Voraussagen, die die Spieler zuvor abgegeben haben.

Broker

Soll eine Aktie im Wert steigen, wählt man den Menüpunkt: “Aktie steigt im Wert mit x-prozentiger Wahrscheinlichkeit“. Das Programm stellt verschiedene Prozentwerte zur Verfügung. Je höher dieser Wert gewählt wird, desto mehr Punkte muß der Spieler dafür opfern. Zu Beginn stehen jedem Mitspieler 1000 dieser Punkte zur Verfügung. Das klingt im ersten Moment viel, aber beim Versuch, seine eigenen Aktienkurse anzuheben, sind sie schnell aufgebraucht.

Optisch bietet das Programm naturgemäß wenig. Doch reizt es durch die interessante Thematik und die spannenden Finanzschlachten, die sich zwischen mehreren Mitspielern ergeben können.

Name: Broker
Quelle: Oase

Ohne ein gutes Gedächtnis werden Sie bei Senso wohl keine besonders guten Ergebnisse erzielen. Es geht darum, eine Folge aus verschiedenen Tönen nachzuspielen. Jedesmal, wenn eine Tonfolge vom Spieler komplett richtig reproduziert wurde, wird sie durch eine neue, längere Tonfolge ersetzt.

Senso

Insgesamt gibt es vier verschiedene Töne, die sich in vier verschiedenfarbigen Feldern widerspiegeln. Jedesmal wenn ein Ton ertönt, leuchtet das entsprechende Farbfeld parallel dazu auf. Drehen die Spieler den Ton ab, können anstelle der Tonfolgen die Farbfolgen erraten und reproduziert werden.

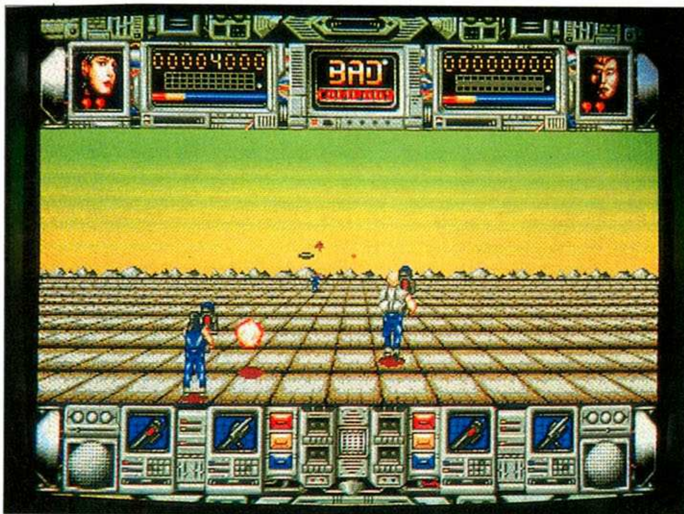
Senso ist ein einfaches, aber vernünftig gemachtes Gedächtnisspiel, das insbesondere in Gesellschaft viel Spaß macht.

Der Autor Peter Händel zeichnete schon in der Vergangenheit für einige andere, wirklich gelungene Spiele verantwortlich (Steinschlag, Marble Slide etc.). Auch er scheint über die Art, wie in der Vergangenheit mit den Programmierleistungen enthusiastischer PD-Autoren Geschäfte gemacht wurden (und nach wie vor gemacht werden), unzufrieden gewesen zu sein. Er bezeichnet sein Programm in aller Deutlichkeit als Freeware und besteht im Vorspann auf eine nichtkommerzielle Weiterverbreitung. Mit Fug und Recht, wie wir meinen.

(Ingmar Reyer/hs)

Name: SENSO
Quelle: Franz
Besonderheit: Freeware





Als Kosmo-Söldner auf Alienhatz, keine weiteren Fragen...

bad Company

Wenn es einen Preis für Ehrlichkeit gäbe, so hätte ihn Bad Company redlich verdient! Was hat dieses Ballerspiel also zu bieten?

Zunächst einmal eine der aufrichtigsten Stories, die das Killen von Aliens rechtfertigt: Die Erde braucht neuen Lebensraum, die (ohnehin insektoiden) Aliens stehen dem im Wege. Also fort mit ihnen – natürlich zum Wohle der Menschheit, gelle? Für die Säuberung steht eine Truppe von acht Söldnern zur Verfügung, aus denen der Spieler seinen Weltraum-Rambo wählen darf. Insgesamt sind vier Planeten zu 'reinigen', auf denen neben Horden von Aliens auch Waffennachschub und Lebensenergie zu finden sind. Stehen genügend Joysticks zur Verfügung, können sich auch zwei Spieler auf die fröhliche Hatz begeben.

Tja, das war's dann auch schon. Das dreidimensionale Szenario scrollt in Fluchtpunktperspektive auf den Spieler zu, der braucht nichts weiter zu tun, als aufs Knöpfchen zu drücken und den gegnerischen Schüssen auszuweichen. Die Steuerung ist brauchbar, die Grafik guter Durchschnitt, nur vom Sound von David Whittaker wünscht man sich mehr (und vielleicht auch Besseres). Da jeder Kämpfer bestimmte Vorzüge hat und das Arsenal an Waffen reichhaltig ist, ist die Motivation anfangs recht hoch – allerdings nur am Anfang. Es sei noch ein Zitat aus dem Covertext gestattet, der wie die Faust aufs (Alien-)Auge paßt: 'Fragen werden hier keine gestellt, und Antworten verhallen im Leeren. Es gibt nur eine Regel: Vernichtung der Aliens.' In diesem Sinne: Feuer frei...

(Michael Anton/hs)

AMIGA DOS Blitzlicht

Name: Bad Company
Hersteller: Logotron Entertainment
Vertrieb: Fachhandel
Preis: 64,95 DM

Positiv:
- ehrliche Story
- Option für zwei Spieler

Negativ:
- zu ehrliche Story
- keine Neuheiten





Packen Sie Taktgefühl und Fairness besser wieder ein, hier wird mit dem Granatwerfer argumentiert

5th GEAR

Autorennen einmal anders – das bietet 5th Gear. Es fragt sich nur, ob diese Art von Rennen wirklich so gut ist...

Vergessen Sie alle Regeln der Fairness, Ihre mühsam absolvierten Fahrstunden – und vielleicht auch die Ansprüche, die Sie an Amiga-Spiele stellen. Denn bei 5th Gear können Sie den Gegner mit MG oder Rakete von der Strecke pusten, mit absolut fehlgeplanten Strecken rechnen – und den Rest getrost vergessen...

Der Hintergrund wird nicht näher erläutert: Sie nehmen an einem illegalen Rennen teil, bei dem alles erlaubt ist, was verfüg- und finanzierbar ist. Diverse Shops auf der Strecke bieten neben dem autospezifischen Service wie Reparatur und Benzin alles, um aus dem Raser einen Kampfpanzer zu machen: Schnellfeuergewehr und Raketenwerfer gibt es, mit Boostern kann man die Karre auch für kurze Zeit zum 'Flugzeug' machen. Mit dieser Ausrüstung geht es dann auf Strecke, eine in der Vogelperspektive dargestellte Wüstenlandschaft, in der es von Hindernissen und Gegnern nur so wimmelt.

(Offensichtlich wurde vergessen, das Auto mit einer effektiven Bremse auszurüsten.) Gewinnt man dennoch ein Rennen, gibt es Geld für neue Ausrüstung und eine neue Runde mit neuem Limit.

Die grafische Gestaltung ist akzeptabel, Steuerung und Sound während des Spiels könnten besser sein. 5th Gear ist ein Spiel, welches kurzzeitig Spaß macht, auf längere Sicht aber zu wenig bietet. Schade, denn man hätte mehr aus dieser Idee machen können...

(Michael Anton/hs)

AMIGA DOS Blitzlicht

Name: 5th Gear
Hersteller: Hewson
Vertrieb: Fachhandel
Preis: 64,95 DM

Positiv:
- fairer Preis

Negativ:
- hakelige Steuerung
- stellenweise unfair





Unterwasseraction hat Konjunktur, Aquanaut verfolgt in der Tiefsee Aliens

Aquanaut

Da weder den Aliens noch den Programmierern Neues einfällt, ist die Erde mal wieder in Gefahr. Oder gibt's doch was Neues?

Eher zufällig ist die für das 21. Jahrhundert geplante Invasion der bösen Ramanishi schief gegangen. Da so was keine gute Story abgibt, haben die Herren von Prism wenigstens einen Alien auf die Erde gelangen lassen, damit zumindest etwas zu machen ist... Besagter Alien hat sich inzwischen auf dem Meeresgrund niedergelassen und macht dort – ja, was macht er dort? Das weiß keiner so recht, es wird auch aus der Story nicht klar. Aber rein prophylaktisch sollte man ihm wohl mal auf die Finger klopfen, vielleicht erfährt man dabei ja etwas Neues. Seinen Kopf hierfür muß ein Marinetaucher herhalten, der ins eiskalte Wasser gestoßen wird. Dort findet er neben Kapseln mit Ausrüstung auch jede Menge mutierter Meeresfauna, die ihm ans Leder will. Durch drei Szenarien muß er sich quälen, bis die Menschheit gerettet ist: Das offene Meer, ein Labyrinth unter Wasser und eine geheimnisvolle Stadt warten auf ihre Erforschung...

Grafik und Sound könnten manchmal etwas besser sein, alles in allem besitzt das Spiel jedoch seine Reize.
Sehr lobenswert ist auch, daß eine einmal gelöste Sequenz nicht immer neu gespielt werden muß. (Weniger gut ist, daß darüber auf der Originaldiskette Buch geführt wird, Paßwörter wären zur Virenabschreckung doch besser gewesen.) Schade auch, daß der Rechner nach Spielende abgeschaltet werden muß, ein Reset ist nicht möglich...

(Michael Anton/hs)

AMIGA DOS Blitzlicht

Name: Aquanaut
Hersteller: Prism
Vertrieb: Fachhandel
Preis: 84,95 DM

Positiv:
- Spiel komplex und abwechslungsreich
- Zwischenstand speicherbar

Negativ:
- Grafik und Sound etwas dürftig
- Preis etwas hoch





Auch bei diesem Fußballspiel kann es zu spannenden Zweikämpfen kommen

Bodo Ilgners Super Soccer

So manchen packt bei der Übertragung eines heißen Bundesligaspiels die Lust, einmal selbst dem runden Leder nachzujagen. Und damit man sich nicht überanstrengt, gibt es die Fußballsimulationen.

Fußballspiele wurden schon oft und in allen Variationen auf Heimcomputern realisiert. Da gab es zum Beispiel Programme, in denen einfach nur gespielt wurde oder andere, in denen der Spieler die Rolle des Managers übernahm, und Mischungen aus beidem. Bodo Ilgners Super Soccer gehört zu letzteren. In den Menüs können Sie sich eigene Mannschaften, Ligen und Pokale entwerfen oder Ihr Team in verschiedenen Situationen trainieren. Außerdem hat jeder Spieler Eigenschaften wie Können, Kraft, Charakter und Spielstil und ein von Ihnen veränderbares Outfit wie Haarlänge und Farbe, Teint und Name.
Im eigentlichen Spiel können zwei Spieler gegeneinander oder ein Spieler gegen den Computer spielen. Beim Schießen geben sogenannte Boot-o-Meter Schußstärke und Höhe an.
Viel Verwirrung verursacht die Darstellung des Spielfeldes: Befindet sich der Ball im Mittelfeld, so wird der Platz in der Seitenansicht gezeigt. Kommt er allerdings in Tornähe, so wird das Tor von vorn gezeigt, so daß man nun von unten nach oben spielt. Kommt der Ball dann wieder ins Mittelfeld, geht's wieder seitwärts weiter.
Bodo Ilgners Super Soccer gehört eher zu den schlechten Vertretern seines Genres, denn weder Grafik, Sound oder das Spielgeschehen können überzeugen.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Bodo Ilgners Super Soccer
Hersteller: Empire Software
Quelle: Fachhandel
Preis: zirka 84,95 DM

Positiv:
- viele veränderbare Parameter
- eigene Ligen und Pokale möglich

Negativ:
- wechselnde Perspektive
- schwache Grafiken



AMIGA DOS-Spieletips

Viele heiße Tips für harte Spiele haben wir aus unserer großen Kiste mit den Spiellösungen herausgesucht und für diese Helpline zusammengestellt. Auch diesen Monat können wir dank der freundlichen Unterstüzung durch Sie, liebe Leser, wieder viele Spielrätsel aufklären. Wenn Sie sich an der Helpline beteiligen wollen, auch Ihre Tips sind willkommen und werden bei der Veröffentlichung mit einem Software-Dankeschön honoriert.

Kennen Sie den liebenswerten und etwas tolpatschigen Helden aus dem Abenteuer "Future Wars – Time Travellers"? Der hat nämlich den ersten Teil seines schweren Abenteuers hinter sich und befindet sich zur Zeit irgendwo in der Zukunft. Bevor er sich an die Erkundung der für ihn unbekannten Gegend macht, beschließt er zunächst einmal, seine bisherigen Erlebnisse für die Nachwelt in einem Tagebuch festzuhalten. In der AMIGA DOS können Sie seine Eintragungen lesen.

Zukunftskriege – die Abenteuer eines Fensterputzers

"...Die ganze Geschichte nahm beim Fensterputzen ihren Lauf. Ich weiß noch, wie ich, ausgerechnet als mein Boß aus dem Fenster schaute, den Putzeimer mit dem Fuß umstieß. Auf dieses Mißgeschick folgte eine lange Predigt seinerseits. Dabei regte er sich so sehr auf, daß er sein Fenster nicht richtig schloß. Ich nahm also meinen Eimer und schaute mir meine Plattform etwas genauer an. Auf dem Schaltpult drückte ich den Knopf für "rauf" und fuhr auf die Etage mit dem nicht fest verschlossenen Fenster. Nach dem Öffnen war der Weg ins Innere des Gebäudes frei. Eigentlich wollte ich nur meinen Eimer im Bad wieder füllen, aber einmal hier eingedrungen, beschloß ich, mich in der Chefetage etwas genauer umzusehen. Als erstes fiel mir der Papierkorb ins Auge. Er enthielt eine leere Plastiktüte, die ich in weiser Voraussicht an mich nahm. Im Bad wurde der Eimer am Waschbecken gefüllt. Im Schrank stand eine Flasche mit Insektizid, die in meinen Besitz überging. Im WC fand ich dann noch ein kleines rotes Fähnchen. Nachdem ich alle Türen wieder verschlossen hatte, durchquerte ich den Raum, um ihn durch die Tür nach rechts zu verlassen. Auf meinem Weg stolperte ich über eine Beule im Teppich. Die Ursache dafür war ein kleiner Schlüssel, der darunter lag. Dann bekam ich ein Problem: Jedesmal, wenn ich die Tür auch nur berührte, öffnete sich die Tür meines Chefs, und er ließ ein erneutes Donnerwetter los. Ich tat dann lieber so, als wollte ich ganz schnell wieder verschwinden. Nach einiger Zeit kam mir eine geniale Idee: Ich positionierte meinen vollen Eimer direkt über seiner Tür. Von Vorfreude gepackt machte ich mich auf den Weg zur rechten Tür. Gleich nach dem Öffnen, trat auch der Boß aus seinem Zimmer und der Eimer traf sein Ziel exakt. Schnell huschte ich ins nächste Zimmer. Ich schloß die Tür und nahm die Einrichtung in Augenschein. Einer der Schränke, der sich mit dem Schlüssel, den ich unter dem Teppich gefunden hatte, öffnen ließ, enthielt eine Schreibmaschine, mit deren Carbonfarbband eine Nummer geschrieben wurde, die ich mir glücklicherweise merkte. In der Schreibtischschublade habe ich dann noch einen Stapel Papier gefunden. Als nächstes nahm ich die Wandkarte in Augenschein, auf der mir besonders ein kleines Loch auffiel, in das das rote Fähnchen paßte. Darauf gab die Karte eine Geheimtür frei.

Neugierig betrat ich den fremd anmutenden Raum dahinter. Ich bemerkte mit Entsetzen, daß sich nach meinem Betreten die Decke langsam senkte. Die einzige Hoffnung bestand in der Tastatur an der Wand, mittels derer ich so schnell wie möglich die Nummer aus der Schreibmaschine eingab. Welch ein Glück, daß mein Einfall richtig war! Die Decke hob sich wieder und die Wand vor mir gab einen Durchgang frei, hinter dem sich ein Raum mit seltsamen Geräten befand. In der Mitte des Raumes stand eine Maschine, die wie ein Kopierer aussah. Ich drückte auf den grünen Knopf und die Maschine erwachte zum Leben, was sich in einem tiefen Brummen äußerte. Einer Inspiration folgend schob ich das Papier in den Schlitz an der Seite der Maschine. Dann drückte ich den roten Knopf, nahm die Dokumente, die die Maschine wieder ausspuckte und ging durch das Licht des Deckenstrahlers. Der Raum verschwand, und ich fand mich in einem Sumpf wieder. Das Vorankommen war sehr schwer, da die Stücke festen Bodens nur sehr schmal waren. Gegen die Moskitos kam das Insektenspray zum Einsatz. Am Ende des Sumpfes lag ein Amulett auf dem Boden. Ich kam an ein wunder-



schönes Seeufer, und nachdem ich mich etwas umgeschaut hatte, entdeckte ich ein Seil in einem Loch am Fuße des Baums links im Bild. Ich warf das Seil über einen Ast des Baums und kletterte daran hinauf.

Oben wartete ich eine kleine Weile, und bald erschien ein seltsam gekleideter Mann, der sich auszog und im See badete. Die Gelegenheit ausnutzend, zog ich seine Kleider an. Mein weiterer Weg führte mich nach links, und ich stand vor einem großen Schloß. Mit Erstaunen stellte ich fest, daß hier alles mittelalterlich aussah. Der Wache, die vor dem Schloß stand, zeigte ich das Amulett.

Sie versprach mir, ihrem Herrn davon zu berichten, sobald dieser zurück sei. Um mir die Zeit zu vertreiben, ging ich um das Schloß herum, und bald tat sich eine Lichtung vor mir auf. In der Mitte stand ein großer Baum, über dessen Ast die Kutte eines Mönches hing. Ich schüttelte den Baum, mit dem Erfolg, daß sich eine Münze aus den Falten des Gewandes löste und auf den Boden fiel.

Nach einigem Suchen fand ich sie dann auch. Mit der Münze ging ich in das Wirtshaus. Sie verhalf mir zu einer Mahlzeit, mit der ich einige Zeit überbrücken konnte. Dabei hatte ich Gelegenheit, den Gesprächen am Nachbartisch zuzuhören. Den Worten entnahm ich, daß die Mönche, die hier in einem Kloster seßhaft waren, sich in letzter Zeit mehr als seltsam benahmen.

Als ich genug gehört hatte, verließ ich das Wirtshaus und machte mich erneut auf den Weg zur Wache. Nun war der Herr des Hauses anwesend.

Ich trat ein und zeigte ihm das Amulett. Er erzählte mir, daß es seiner Tochter gehöre, die aber verschwunden sei. Leichtsinnig, wie ich nun einmal bin, versprach ich dem alten Mann, seine Tochter zurückzubringen.

Draußen schlief die Wache auf ihrem Posten. Die Gelegenheit ausnutzend, entwand ich ihr ihre Lanze. Mit Hilfe der Lanze konnte ich die Kutte vom Ast holen. Nachdem ich sie angelegt hatte, sah ich aus wie ein waschechter Mönch. Ich ging vor dem Schloß nach Süden und kam an eine Brücke, die von einem großen Wolf bewacht wurde. Ein elektrisches Kabel, das aus seinem Bauch ragte, brachte mich auf eine Idee: Ich ging zum See und füllte die Tüte mit Wasser. Anschließend begab ich mich auf dem schnellsten Wege wieder zu dem Wolf und warf die volle Tüte über den elektrischen Gesellen, der darauf in einem grellen Blitz verpuffte. Nun war der Weg in das Kloster frei.

Im Kloster gab es einen Raum, in dem sich Mönche ständig rechts herum im Kreis bewegten. Alle Bewegungen, die ich in dieser Halle machte, tat ich also in dieser Richtung und achtete darauf, nicht durch die Mitte des Raumes zu gehen. Mein erstes Ziel war die westliche Tür. Hier gab mir ein Mönch den Auftrag, dem Obervater eine Nachricht zu überbringen. Um nicht aufzufallen, nahm ich diesen Auftrag an. Der Vater befand sich in einem Raum hinter der östlichen Tür. Er verlangte nach einem Becher Wein. Das einzige Behältnis, das sich dazu eignete, befand sich im westlichen Raum. Glücklicherweise war er menschenleer, so daß ich den Kelch an mich nehmen konnte. Hinter der nördlichen Tür befand sich ein Weinkeller, in dem ich den Becher füllen konnte. Mit dem vollem Becher begab ich mich zum Vater. Nach einigen Schlucken war dieser total betrunken und schlief ein. Eine gründliche Untersuchung des Mönchs brachte eine Fernsteuerung zutage, mit der man eine Geheimtür in dem Möbelstück öffnen konnte, hinter der eine Magnetkarte lag. Mit diesen Gegenständen begab ich mich dann wieder in den Weinkeller, wo sich mit der Fernsteuerung das oberste Faß öffnen ließ und einen Weg in ein Labor freigab. Hier war auch die Tochter des Burgherren. Die Karte paßte in den Schlitz eines Computers, und das hübsche Mädchen wurde auf diese Weise befreit. Das Amulett wies mich als Freund aus. Sie tippte eine Sequenz in den Computer ein und aktivierte die Selbstzerstörung der Anlage und einen Beamer. Kurz darauf befanden wir uns im Schloß, wo mir eine seltsame Geschichte erzählt wurde. Auf jeden Fall stammten die beiden aus der Zukunft und sie wollten mich aus Dank mit dorthin nehmen. Irgend etwas muß aber beim Transport schiefgelaufen sein, denn ich kam allein in einem riesigen Ruinengebiet an. Nach Vollendung dieser Zeilen werde ich mich auf den Weg machen und die Gegend erforschen."



Soviel zu Future Wars. Vielleicht können wir in einer der nächsten Ausgaben einen weiteren Teil seines Tagebuches veröffentlichen.

Wir bleiben zeitlich gesehen in der Zukunft, denn unsere nächsten Tips, die uns Stefan Stockinger aus Hollabrunn in Österreich zusandte, beziehen sich auf Space Ace, dem Nachfolger von Dragons Lair. Mit Hilfe der Tips sollte es nicht mehr schwierig sein, Commander Borf von seinen teuflischen Plänen abzuhalten.

Space Ace

Die Tips beschränken sich auf Anweisungen, wann der Joystick in welche Richtung gedrückt werden muß. Zur besseren Orientierung haben wir jeweils eine kurze Beschreibung der Szenen beigefügt. Manchmal ist der Name einer Szene allerdings selbsterklärend.

1. Szene "Laserbeschuß":
Space Ace wird von Borf mit einem Laser beschossen. Kurz warten, dann *rechts, links, runter*

2. Szene "Roboter I":
Ein Roboter zerstört den Boden. Sofort *links*, wenn der rechte Arm oben ist *rechts*, dann warten, bis der linke Arm runter und wieder rauf ist, *links*, wenn der linke Arm wieder oben ist *links*

3.Szene "Ufos":
Ace wird von Ufos beschossen. *Runter* und kurz warten, dann *rauf*

4. Szene "Flug":
Ace landet auf der Station. Kurz vor der Landung: *rauf*

5. Szene "Schlange":
Ace wird von einer auftauchenden Schlange angegriffen: *Feuerknopf*



6. Szene "Stampfer":
Ace steht auf einer Brücke, die von einem Stampfer zerstört wird. *Rechts*, solange wie möglich warten, dann *rauf*

7. Szene "Dose":
Ace steht vor einer "Dose", die sich auf- und abbewegt. Sobald die Dose ganz unten ist *rechts*, sofort nach der Landung *rechts*

8.Szene "Monster I":
Ace wird von einem Monster von links angegriffen. Kurz vor dem Monster *runter, rechts*

9. Szene "Monster II":
Ace wird von einem Monster von rechts angegriffen. Kurz vor dem Monster *runter, links*

10. Szene "Monster III":
Ace wird von einem Monster von links angegriffen: *Feuerknopf*

11. Szene "Vor dem Schloß":
Wenn die Katzen am unteren Rand erscheinen, *rauf*

12. Szene "Hunde I":
Ace wird von Hunden verfolgt: *rauf*

13. Szene "Hunde II":
rechts

14. Szene "Hunde III":
rauf

15. Szene "Roboter II":
Ace steht zwischen zwei Robotern, die auf ihn schießen wollen: *Rechts*

16. Szene "Feuerstrahl I":
links

17. Szene "Feuerstrahl II":
links

18. Szene "Feuerstrahl III":
rechts

19.Szene "Leiter":
rauf

20. Szene "Kampf I":
Abwehr, *Feuerknopf*

21. Szene "Kampf II":
Abwehr, *Feuerknopf*

22. Szene "Kampf III":
Abwehr und nach rechts rollen, *Feuerknopf, rechts*

23. Szene "Kampf IV":
Feuerknopf, runter

24.Szene "Kampf V":
Abwehr, *Feuerknopf*

25. Szene "Kampf VI":
Ausweichen, *rauf, runter*

26.Szene "Kampf VII":
Ausweichen, von hinten anspringen, *rechts, runter*

27. Szene "Huckepack I":
keine Aktion

28. Szene "Huckepack II":
die Naheinstellung, sofort *links*

29.Szene "Plattform":
keine Aktion

30. Szene "Lavabecken":
rechts

31. Szene "Schuß I":
keine Aktion

32. Szene "Einschlag I":
dem Schuß ausweichen, *rechts*

33. Szene "Schuß II":
keine Aktion

34. Szene "Einschlag II":
links

35. Szene "Schuß III":
keine Aktion

36. Szene "Einschlag III":
rechts

37. Szene "Spiegel":
Spiegel schieben und zurückspringen, *links, rechts* – *geschafft*

Es folgen noch einige Schlußanimationen, dann ist das Spiel zu Ende.

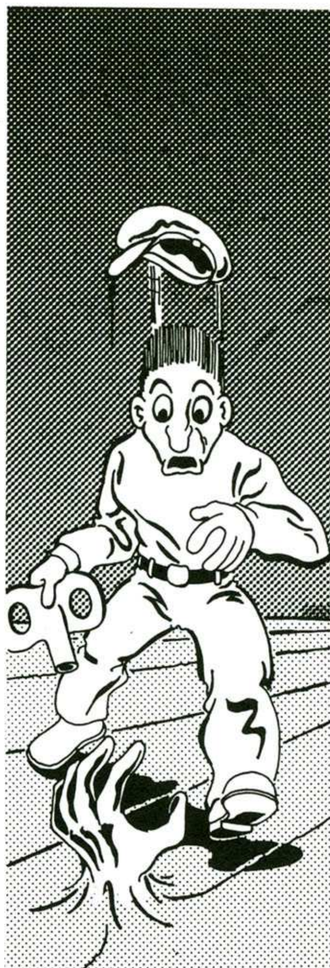
Zak McKraken von Lucasfilm Games erfreut sich nach wie vor einer großen Beliebtheit. Aus diesem Grund haben wir uns entschlossen, den Lösungsweg zu diesem Abenteuer in unsere Helpline aufzunehmen.

Zak McKracken – Im trauten Heim

In Zaks Wohnung befinden sich eine Menge Gegenstände, die sich später für uns noch einmal als nützlich erweisen könnten. Zunächst wäre da Sushi samt Fischglas, der in unsere Tasche wandert. In der Schrankschublade liegt eine Telefonrechnung, die ebenfalls eingesteckt wird. Apropos Telefon: Um auch während unserer Abwesenheit immer auf dem Laufenden zu sein, schalten wir den Anrufbeantworter ein. Als nächstes fällt uns ein häßlicher Tapetenfetzen an der Wand auf, der abgerissen und mitgenommen wird. Wenden wir uns nun dem Schreibtisch zu. In der Schublade liegt ein Kazoo und unter ihm unsere Cashcard. Pech, daß wir nur schlecht herankommen und sie statt aufzuheben nur tiefer unter den Tisch schieben. Doch mit der Telefonrechnung wird die Aktion zum Kinderspiel. Soviel zum Thema Schlafzimmer. Unser nächster Weg führt uns ins Wohnzimmer. Das Sofa ist unaufgeräumt – das linke Kissen liegt nicht ordentlich an seinem Platz. Beim Aufräumen stellen wir fest, daß die Fernsteuerung darunter liegt. Hinter dem Sofakissen, das wir soeben von der Wand genommen haben, bemerken wir, daß der Stecker des Fernsehers nicht an seinem Platz steckt: Dem kann man abhelfen. Nachdem nun der Fernseher mit Strom versorgt ist und sich auch die Fernbedienung in unseren Händen befindet, liegt nichts näher, als ihn einzuschalten. Hier können wir zum ersten Mal Annie Larris bewundern. Der Sendung kann man einige wichtige Hinweise auf den weiteren Spielverlauf entnehmen. Weiter geht es in die Küche. Wir bringen das Buttermesser in unseren Besitz. Es hängt über dem Spülbecken. In dem Schränkchen unter dem Spülbecken befindet sich eine Schachtel mit Buntstiften. Auch das Ei aus dem Kühlschrank findet seinen Weg in unsere Tasche. Wenn wir jetzt noch den kleinen Schlüssel neben der Tür an uns nehmen, können wir die Wohnung verlassen.

Frisch ans Werk

Draußen wenden wir uns nach links und gehen zum Bäcker. Nach dreimaligem Klingeln ist er so verägert, daß er mit einem steinharten Brot nach uns wirft. Das läßt uns aber kalt, und wir nehmen das Brot an uns. Unser nächstes Ziel ist das TPC-Gebäude, in dem wir uns eine Anmeldung abholen. Sie befindet sich in der Apparatur links hinten an der Wand. Also gehen wir zurück zu unserem Haus, füllen die Anmeldung mit dem gelben Stift aus und legen sie in den Briefkasten, nachdem wir ihn mit dem kleinen Schlüssel geöffnet haben. Als nächstes führt uns unser Weg nach rechts in die 14th Avenue, in der wir das LL-Gebäude betreten. Hier kaufen wir alles, bis auf die Gitarre (den Wet Suit ganz links nicht vergessen!). Weiter nach rechts zum Friseur. Wir öffnen die eben erworbene Werkzeugkiste und schneiden mit dem "wire cutter" das "bobby pin sign" (Haarnadelschild) ab. Nun geht es wieder nach Hause. In der Küche schrauben wir das Abflußrohr mit der "monkey wrench" ab und stecken das Brot in das Spülbecken.



Nach einem Druck auf den Knopf über dem Herd verwandelt sich das Brot in Krumen, die man unten einsammeln kann. Wir verlassen die Wohnung wieder und begeben uns erneut ins TPC-Gebäude, aber nicht ohne uns vorher mit Hut und Brille getarnt zu haben. An dem Computerterminal (erst die Thekentür links aufmachen) können wir dann die Telefonrechnung begleichen (use computer terminal). An dieser Stelle werden wir unseren geliebten Fisch Sushi los, indem wir in liebevoll, aber mitleidslos in den Kübel der Topfpflanze gießen. Das Glas behalten wir aber nach wie vor. Wieder auf der Straße ziehen wir zunächst den Hut und die Brille ab und begeben uns dann zum Bus. Nachdem wir den Busfahrer sanft mit unserem Spiel auf dem Kazoo aus dem Schlaf gerissen haben, öffnet er uns ungern die Tür. Wir treten ein, stecken die Cashcard in den dazugehörigen Reader und fahren in Richtung Flugplatz.

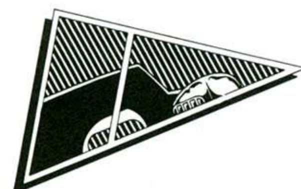
Wenn einer eine Reise tut...

Auf dem Flughafen kaufen wir einem Sektenmitglied ein Buch ab, damit wir auf der langen Reise auch etwas zu lesen haben. So ausgerüstet können wir das Flugzeug aufsuchen. Um im Flugzeug in Ruhe agieren zu können, müssen wir unbedingt die Stewardeß beschäftigen. Dazu gehen wir zunächst ins Heck des Flugzeugs auf die Toilette. Hier stecken wir das Klopapier in den Ausguß des Waschbeckens und drehen den Wasserhahn auf. Wenn das Wasser überläuft, drücken wir die Ruftaste über dem Waschbecken. Jetzt ganz schnell nach rechts zur Mikrowelle und das Ei hineingelegt, die Mikrowelle schließen und anschalten. Von diesem Moment an kann man sich dann wieder Zeit lassen.

Wir untersuchen den ersten Sitzplatz etwas genauer, unter dessen Polster wir ein Feuerzeug finden. Nun untersuchen wir noch die Gepäckfächer. In einem (natürlich dem letzten, egal welches das ist) finden wir eine Sauerstoffmaske, die in unseren Besitz übergeht. Jetzt können wir sowohl den restlichen Flug als auch das Gezeter der Stewardeß genießen.

In Seattle

Nachdem wir den Flug mehr schlecht als recht hinter uns gebracht haben, brechen wir in Seattle einen Ast von einem Baum ab und füttern ein Eichhörnchen mit zwei Köpfen mit den Erdnüssen, die wir von der Stewardeß bekommen hatten. Nun können wir mit dem Messer den Dreck von der Felswand abkratzen und legen so den Eingang zu einer Höhle frei. Im nächsten Bild herrscht Finsternis, die man aber mit dem Feuerzeug ein wenig durchbrechen kann. In der Mitte der Höhle befindet sich ein Nest, das wir mit Hilfe des Golfschlägers herunterholen können. Jetzt tasten wir uns zur Feuerstelle voran, in die wir sowohl das Nest als auch den Ast legen. Wenn wir beides mit dem Feuerzeug entzünden, wird es hell und warm in der Höhle, so daß wir uns wesentlich besser orientieren können. Ganz rechts befinden sich seltsame Schriftzeichen an der Wand, die wir mit dem



Stift zu dem Symbol aus unserem Traum ergänzen. Dadurch öffnet sich für uns der Weg in eine Kammer. An den blauen Kristall kommen wir mit Hilfe der Fernsteuerung. Nach getaner Arbeit begeben wir uns zurück zum Flughafen und kaufen uns ein Ticket zurück nach Frisco.

Wieder zu Hause

Als erstes entnehmen wir dem Briefkasten die King Elvis Fanclubkarte (small key benutzen). Ist sie noch nicht angekommen, weil wir zu schnell waren, erledigen wir erst die anderen Dinge hier in San Francisco. Aber wir dürfen sie auf gar keinen Fall hier vergessen! Gehen wir in die 14th Avenue und werfen den blauen Kristall in den Schlitz der roten Tür, worauf uns Annie die Tür öffnet und in die Wohnung bittet. Nun erhalten wir von Annie den ersten Teil des gelben Kristalls und können von nun an zwischen Annie, Zak, Leslie und Melissa hin- und herschalten. Diese Fähigkeit probieren wir auch gleich einmal aus. Wir schalten auf Annie um und nehmen die Cashcard, die unter der Schreibunterlage liegt. Jetzt schalten wir wieder auf Zak und buchen am Flughafen einen Flug über London nach Katmandu.

In Katmandu

In Katmandu zeigen wir dem Wächter das Buch der Erleuchtung, worauf dieser uns das Haus betreten läßt. Jetzt steht uns der Weg zum Guru offen. Der läßt uns allerdings noch einige Zeit warten, da wir wegen Sushi ein schlechtes Karma haben, das erst noch verfliegen muß. Ist dies geschehen, erklärt uns der Guru, wie wir uns mit dem gelben Kristall in Tiere verwandeln können. Wieder draußen, zünden wir das Heu mit dem Feuerzeug an. Dies beschäftigt die Polizisten so sehr, daß wir jetzt die Fahne samt Stange vor dem Gefängnis entwenden können. Von dem Yak lassen wir uns für Geld zum Flughafen transportieren. Von hier fliegen wir über London nach Miami.

In Miami

In Miami tauschen wir mit dem Penner das Buch gegen eine Flasche Whisky und fliegen über Mexico City nach Lima.

In Lima

Im Dschungel von Lima suchen wir eine Lichtung mit einer Steinplatte. Von jetzt an ist Eile geboten, da wir sonst von einem Alien gefangengenommen werden. Wir legen also die Brotkrumen auf den Stein, warten, bis der Vogel kommt und verwandeln uns in ihn. Als Vogel fliegen wir nach rechts in das linke Auge der Felsfigur, nehmen den Scroll mit und fliegen wieder zurück. Wir übergeben den Scroll an Zak und verwandeln uns wieder zurück. Jetzt nichts wie weg in den Dschungel. Zurück kommen wir jetzt entweder mit dem Flugzeug oder, was wesentlich billiger ist, wir lassen uns bei dem Stein von einem Alien gefangennehmen, der uns zurück nach San Francisco bringt.

Die Befreiung

Falls wir also einmal von den Aliens gefangengenommen worden sind, können wir uns ganz einfach wieder befreien. Die Aliens sperren uns in die Verdummungsmaschine und verschwinden. Hier beginnen unsere Befehlcodes langsam zu verschwinden, also müssen wir uns beeilen. Wir brauchen eigentlich nur den Hut und die Brille aufzusetzten. Nach einiger Zeit kommt dann ein Alien und läßt uns frei, da es uns für einen der ihren hält. Aus dem Schrank können wir dann unsere gesamte Habe an uns nehmen und wieder auf die Straße gehen, wo wir Brille und Hut wieder absetzen. Sollten, aus welchen Gründen auch immer, sich Brille und Hut nicht mehr in unserem Besitz befinden, warten wir einfach, bis wir keine Codes mehr haben und lassen uns von den Aliens auf die Straße setzen, wo die Codes dann langsam wiederkehren. Vom Haus aus kommt man durch einen Geheimgang, der unter einer Bodenplanke beginnt, in das Zimmer mit der Maschine, wo wir unsere Sachen wieder an uns nehmen können.

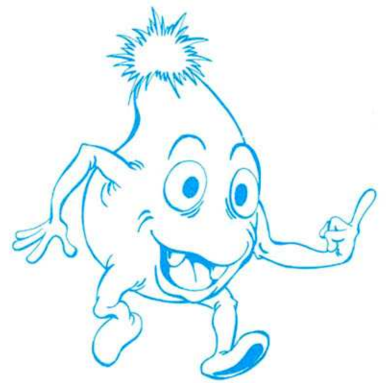
Wie es mit Zak, Annie, Leslie und Melissa weitergeht, und was sie noch alles erleben, können Sie in unserer nächsten Ausgabe nachlesen. Wir fahren einstweilen mit der Lösung zu einem Sierra-Spiel fort.

Die Lösung von MANHUNTER NEW YORK

Für alle Menschenjäger hier die Komplettlösung des ersten Teils der neuen Sierra-Serie.

Der erste Tag

Als erstes werden, wie auch an den folgenden Tagen, alle suspekten Personen mit dem Tracker verfolgt, damit alle Schauplätze erreichbar sind.



Im BELLEVUE HOSPITAL werfen wir einen Blick auf die Leiche und geben ihren Namen ins MAD ein. Der nächste interessante Schauplatz ist die BROOKLYN BAR. Das dortige Videospiel können wir erst nach erfolgreichem Messerwerfen begutachten. Die Bewegung des Mannes nach unserem Erfolg merken wir uns. Beim Videospiel empfiehlt es sich, das Labyrinth im Gedächtnis zu behalten, ebenso sollte es einmal auf dem kürzesten Wege durchlaufen werden. Die Reihenfolge, in der die Figuren dabei abgeschossen werden, gewinnt später noch an Bedeutung. (Siehe auch Karte in AMIGA DOS 2/90.)

Als nächstes suchen wir einen bestimmten Sitz im RESTROOM des PROSPECT PARKs auf. Nach dreimaliger Betätigung der Spülung gelangen wir in die unterirdischen Kloaken. Der Aufbau der Kloake entspricht dem Labyrinth im Videospiel der Bar. An den Stellen, an denen im Spiel Bälle abgeschossen werden, schwimmen Key Cards im Wasser, die wir aufsammeln – insgesamt müssen es zwölf Stück sein. (Es empfiehlt sich, eine Karte des Video-

spiels zu zeichnen und die Positionen der Felder zu numerieren – dadurch kann man gezielt zwischenspeichern, falls man sich mal verlaufen sollte...) Am Ende des Weges finden wir auf dem Steg ein Medaillon, welches wir aufnehmen.

Das nächste Ziel ist CONEY ISLAND. Dort wählen wir die mittlere Schießbude und schießen die Figuren genau wie im Videospiel ab. Gegen Vorlage des Medaillons erhalten wir eine Data Card, die im MAD folgende Informationen offenbart: der Name 'Phil', offensichtlich ein Verräter, und der Hinweis auf eine ominöse 'Lady', die zerstört werden soll. Die nun erscheinende Frage nach dem Verdächtigen können wir beliebig beantworten; der erste Tag ist somit bereits beendet.

Der zweite Tag

Im VEND-O-DELI fällt uns auf einem der Pinboards eine Nachricht auf, die von einer 'Lady' handelt und mit 'Harvey' unterzeichnet ist.

Im CENTRAL PARK folgen wir dem Weg der Verdächtigen, indem wir uns anhand gewisser Landmarken (z.B. nach links zwischen einem roten und einem grünen Baum durch) orientieren. An zwei Stellen wuseln die Personen etwas konfus durch die Gegend. Zuerst müssen wir die entsprechende Stelle im Gebüsch finden, wo ein Brecheisen (Crowbar) herumliegt. Haben wir dieses, geht es ein Bild zurück und dann den Tracker-Weg weiter. Am Ende des Weges liegt eine Leiche, von der wir einiges erfahren können. Bei näherem Hinsehen stoßen wir auf die Namen 'Anna' und 'H. Osborne'. Die Kombinationen 'Anna Osborne' und 'Harvey Osborne' lassen unser MAD Erfolg melden. Die Zeichen auf dem Stein scheinen ein Name zu sein. Mit dem 'P' auf der Stirn der Leiche kommt der Name 'Phil' ins Gedächtnis, und unsere Kombinationsgabe bringt mit 'Phil Cook' einen weiteren Namen ins Spiel, wie das MAD bestätigt.

Der WRETCHED EXCESS NIGHT CLUB ist nur über den Hintereingang zugänglich, der jedoch von vier Schlägern bewacht wird. Die ersten drei gehen ja noch, aber der vierte ist kernig! (Ein Hinweis: an dieser Stelle mit Ton spielen! Es erleichtert die Sache erheblich.) In der Bar sollte auf alle Fälle das Spiel gespeichert werden. Jetzt stehen einige Personen zur Ansicht. Suchen wir uns die richtige aus (ein Tip: Ihre Kutte hebt sich farblich ab), so erhalten wir zwar eines auf die Nase, aber auch eine dreizehnte Key Card und einen Rausschmiß.

In der 152 WEST 82ND STREET, dem Heim von Anne Osborne, finden wir deren Leiche und einen Schlüssel in der Einkaufstasche. Beim MUSEUM öffnen wir die äußere Tür mit dem Schlüssel und folgen dem Weg des Verdächtigen. Die übrigen Türen öffnen wir mit den Key Cards, an der letzten Tür setzen wir das Brecheisen ein. Den daraufhin erscheinenden 'Wachhund' besänftigen wir mit dem Medaillon und gehen weiter. Bei der Leiche im Museum finden sich einige neue Hinweise: Die Zeichnung an der Wand zeigt die Lage eines Gegenstandes in einem Raum. Beim Toten selbst finden wir eine merkwürdige Tätowierung und einen Gegenstand namens Data Module B. Hiermit haben wir das Soll für den zweiten Tag erfüllt und werden wieder abberufen.

Der dritte Tag

In der TRINITY CHURCH zünden wir die linken Kerzen in dem Muster an, das auf der Tätowierung zu sehen war. Es öffnet sich ein Geheimfach, in dem das Data Module A sowie einige Symbole zu finden sind. Auf dem FRIEDHOF merken wir uns den Namen 'Jones' auf dem Grabstein.

Im PAWN SHOP erhalten wir einige Badges zur Auswahl, die wir mit den Symbolen aus der Kirche im Hinterkopf auswählen sollten. Nun stehen wir vor einigen Türen, die sich nur nach Lösung einiger Bilderrätsel öffnen. (Die Lösungen sind 41, 1031, 246 und 425.) Am Ende des Ganges finden wir eine Leiche mit dem Namen 'Harry' auf der Kutte. Eine Ecke weiter lauert schon der böse Phil mit seinem Messer. Haben wir ihn in die Flucht geschlagen, hinterläßt er einen Zettel mit einer wichtigen Zahl.

An der Oberfläche testen wir den Namen 'Harry Jones' im MAD. Beim Besuch seiner Wohnung in der PEARL STREET kommt uns die Zeichnung im Museum in Erinnerung und wir schauen uns das Radio genauer an. Mangels Schraubenziehers öffnen wir es mit der Brechstange und nehmen das Data Module C in Empfang. Im THEATER öffnen wir mit den Zahlen des Zettels den Tresor – und finden einen neuen Code, mit dem wir den Computer im EMPIRE STATE BUILDING, der Residenz von Phil Cook, starten können.

Durch ihn erhalten wir wertvolle Informationen über die Organisation der Orbs und können weitere Weichen stellen: Sektor Alpha ist im Krankenhaus angesiedelt. Den dortigen Robot sollte man von 'Special Security' auf 'Hall Patrol' umstellen. Sektor Beta beherbergt die Flotte der Orbs in der Grand Central Station. Sektor Gamma ist in der Statue of Liberty (die 'Lady'?) untergebracht, wie sich zeigt, wenn die dortigen Robots auf Air Defence programmiert werden. (Allerdings sollte man den Schritt rückgängig machen!) Sektor Delta ist das orbsche Rechenzentrum im Empire State Building. Beim Logout werden wir wieder von den Orbs abberufen.

Der vierte Tag

Heute können wir uns das Tracking sparen, da wir der gesuchte Übeltäter sind. Wenn wir den Robot aus Sektor Alpha richtig programmiert haben, können wir das Krankenhaus genauer erkunden. Zwar werden wir vom Wächter gefangen, ein Fenster, mit der Brechstange zu öffnen, bringt jedoch interessante Einblicke. Nachdem die Orbs den Raum verlassen haben, können wir der Maschine das Data Module D entnehmen und über das eingeschaltete Fließband das Krankenhaus wieder verlassen. In die GRAND CENTRAL STATION gelangen wir mit dem Brecheisen. Dort wird das Raumschiff mit den Data Modules komplettiert und gestartet. (Betätigung von Schalter links, Taste Mitte, Taste links, Taste rechts.)

Durch Schleuse und Labyrinth gelangen wir zum Startplatz. Nach Betätigen des rechten Schalters und dem Blick auf die Anzeige beginnt der letzte Luftkampf. Die vier Sektoren der Orbs müssen zerbombt werden, wobei man sich jedoch auch vor Phil in acht nehmen muß. Sind Freiheitsstatue, Bellevue Hospital, Grand Central Station und Empire State Building zerstört (erfolgreiche Abwürfe werden mit dem Bild der zerstörten Örtlichkeit quittiert!), ist das Spiel gewonnen.

Damit haben wir das Ende der Helpline erreicht. Wir hoffen, daß der eine oder andere Tip bei einem Ihrer Spielprobleme helfen konnte.

(M. Anton/R.Marz/hs)







Unheimlich ist es im finsteren Wald. Was wird Sie hier noch alles erwarten?

Dungeon Quest

Ausgeraubt, ohne Waffen und Geld, befinden Sie sich mitten in der Wildnis und wissen nicht mehr weiter. Dies haben Sie einem Teil des Hilferufes zu verdanken, der Sie per Post erreicht hat. Absender war einer Ihrer besten Freunde, und er scheint wirklich in großen Schwierigkeiten zu stecken…

Da Dungeon Quest ein Textadventure ist, sucht man Icons oder ähnliches vergebens. Zu den Texten kommen noch gute Grafiken und einige Soundeffekte, die zwar stereo, aber doch eher nervend sind. Führt man einen Ortswechsel durch, so wird man mit einer neuen Grafik belohnt, die der Fantasie auf die Sprünge helfen soll.

Der Parser gehört zwar nicht gerade der Elite an, aber man kann ganz gut mit ihm arbeiten. Er weiß auf fast jede Eingabe eine einigermaßen vernünftige Antwort. Allerdings weist er einige Eigenarten auf: Will man zum Beispiel etwas untersuchen, erhält man auf die Eingabe "EXAMINE gegenstand" meistens die Antwort "Your search turns up to nothing", "EXAMINE" ohne weitere Zusätze hingegen listet sämtliche Gegenstände in Form von Texten auf.

Die Story ist im Fantasy-Bereich angeordnet und relativ interessant. Etliche bekannte Elemente, wie der skelettierte Fährmann mit dem einzigen Schiff über den unheimlichen See oder hungrige Drachen, sind enthalten. Obwohl bis an den Rand mit Klischees gefüllt, gelingt es der Story doch, einige Spannung und damit Spielmotivation zu erzeugen.

Abschließend ist zu sagen, daß Dungeon Quest ein Text-Grafik-Adventure der Mittelklasse ist. Im mittleren Bereich ist auch der Schwierigkeitsgrad anzusiedeln.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Dungeon Quest
Hersteller: Gainstar
Quelle: Fachhandel
Preis: zirka 85 DM

Positiv:
- Spielatmosphäre
- gute Grafiken
- Anleitung deutsch

Negativ:
- Spiel komplett in Englisch
- schwache Anleitung





Eine ruhige Hand und große Treffsicherheit sind vonnöten, um in den Wettkämpfen bestehen zu können

John Lowes Ultimate Darts

Darts gehört neben Billard zu den beliebtesten Kneipensportarten. Kein Wunder also, daß es auch auf den Heimcomputern Einzug gehalten hat. John Lowes bietet neben der eigentlichen Simulation auch noch einen Modus, in dem es als Anzeigetafel für eigene Wettbewerbe dient.

In Utimate Darts wurden verschiedene Spielvarianten vereinigt. Da gibt es normale Wettbewerbe, an denen bis zu acht Gegner teilnehmen können, und Spiele wie Rund um die Zielscheibe, Darts Fußball und Darts Cricket.

Im Arcademodus bewegt sich der Pfeil, von der Maus gesteuert, in kreisenden Bewegungen über die Zielscheibe. Wenn man dann klickt, wird der Pfeil nach einer kleinen Verzögerung geworfen. Trifft man einen Pfeil, der schon auf der Scheibe steckt, so prallt der neue Pfeil ab und fällt zu Boden. Es gibt auch eine Option Autowurf, bei der der Computer nach einer bestimmten Zeit automatisch wirft, auch wenn man mit dem Zielen noch nicht fertig ist. Schließlich gibt es noch eine Option mit dem Namen Schaukampf. Hier kann man gegen die besten Spieler der Welt antreten, deren spezielles Wurf- und Treffverhalten in den Computer eingegeben wurde.

Darts-Freunde werden ihren Spaß haben und auch den Realmodus für echte Wettkämpfe zu schätzen wissen. Wer diesem Spiel allerdings schon in der Realität keinen Spaß abgewinnen kann, der wird auch durch dieses Programm nicht zum Darts-Fan.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: John Lowes Ultimate Darts
Hersteller: Gremlin
Quelle: Fachhandel
Preis: zirka 65 DM

Positiv:
- Realmodus Anzeigetafel

Negativ:
- nur für Darts-Freunde interessant





Viele Abenteuer muß der kleine Neuroflummi bei seiner Aufgabe, die Akten und Formulare der Galaxis zu retten, bestehen

Star Trash

Wer kennt sie nicht, die Spiele, in denen eine kleine Kugel über schmale Grate von Gebirgen gesteuert werden muß? Bei Star Trash ist die Kugel ein Neuroflummi und muß die verlorengegangenen Akten der galaktischen Verwaltung von einem Gebirgsplaneten retten.

Der Neuroflummi springt und rollt, von Ihnen gelenkt, durch die Landschaft, in der er die verlorenen Akten suchen muß. Sie können zwischen drei verschiedenen Möglichkeiten der Steuerung wählen.

Der Neuroflummi springt also durch die Gebirgslandschaft und sammelt Gegenstände, die er findet, auf. Da wären zum Beispiel Bonbons und die Akten, die Punkte bringen, oder Blubb, der Fisch, der ebenfalls Punkte bringt, aber schon gefundene Schlüssel auffrißt. Es gibt allerdings auch Wesen, die dem Flummi schaden wollen: Staubsauger, die ihn einsaugen, oder riesige Seepferde oder Dämonen, die buchstäblich auf ihm herumtrampeln. Kommt der Flummi also mit einem solchen unfreundlichen Lebewesen in Berührung oder stürzt er in Folge eines Fehlsprunges in einen Abgrund, verliert er Punkte oder eines seiner anfänglich fünf Leben.

Star Trash kann mit bis zu vier Spielern gespielt werden und bereitet gerade durch seine vielen Animationen sehr viel Spaß. Durch immer neue Gags und Extras bleibt die Motivation über lange Zeit erhalten.

Mit Star Trash liegt mal wieder ein Geschicklichkeitsspiel der freundlichen Sorte vor. Friedlich, niedlich und spaßig. Auch wenn an keiner Stelle der Oscar für besondere Leistungen vergeben werden kann, ist ein solches Spiel inmitten blutrünstiger Aliens und bewaffneter Telespielkonflikte eine echte Wohltat.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Star Trash
Hersteller: Rainbow Arts
Quelle: Fachhandel
Preis: 59,95 DM

Positiv:
- viel Spaß
- hübsche Animationen
- verschiedene Steuerungstypen
- Zeitlupenoption zum genauen Springen

Negativ:
- keine Speicheroption





Flugabenteuer über der bezaubernden Inselwelt von Hawaii – Szenario-Disketten machen es möglich

Hawaiian Odyssey

Dienstagabend ist Krimizeit, pünktlich ab 21.45 Uhr beobachte ich die wilden Verfolgungsjagden, die Detektiv Magnum mit seinem Freund T.C. im Hubschrauber rund um Hawaii unternimmt. Ach, wie gerne würde ich mal mitfliegen! Und siehe da, es geht, und zwar mit der Scenery-Disk namens Hawaiian Odyssey!

Der Flight Simulator hat sich nicht nur auf dem PC als die Flugsimulation überhaupt entpuppt, sondern auch auf dem Amiga. Aber immer nur über der Bucht von San Francisco ins Trudeln zu kommen, ist auf die Dauer langweilig. Die neue Scenery-Disk bietet da einiges an Abwechslung, man kann von jeder der sechs Hauptinseln Hawaiis einen Flug unternehmen.

Das Fliegen selbst gestaltet sich wie gehabt, daß der FS-II kein 'Drauflos-Flieger' ist, versteht sich von selbst. Ausgestattet ist das Softwarepaket mit einer Übersichtskarte von Hawaii mit den wichtigsten Flugplätzen, den Koordinaten der Flugplätze und den Radiofrequenzen.

Die Flugszenen entbehren nicht eines gewissen Reizes. Auch wenn die Vektorgrafik immer zu glatt ist und eben Simulation bleibt, bildet sich beim Flug über den Krater eines aktiven Vulkans schon ein mulmiges Gefühl in der Magengegend.

Die 'Hawaiian Odyssey' ist ein Muß für alle FS-II-Besitzer, nicht nur, weil ohne das Hauptprogramm nichts läuft (oder fliegt), sondern auch wegen der vielen neuen Situationen, vor die der 'Heimflieger' gestellt wird. Auf denn, Burschen, die Hawaii-Sonne geputzt...

(jb)

AMIGA DOS Blitzlicht

Name: Hawaiian Odyssey, Scenery-Disk für Flight-Simulator II
Hersteller: Sub Logic
Quelle: Fachhandel
Preis: 59,95 DM

Positiv:
- realistische Darstellung der Hawaii-Inseln und Flughäfen

Negativ:
- teilweise grobe Vektorgrafik





Footballer of the Year – von Fußball kaum eine Spur!



The Cycles: Rasen, bis der Motor qualmt…

Footballer of the Year 2

Sport ist Mord, hier ist ein Beispiel dafür, wie man mit Sportspielen den Ruf eines Computers und die Nerven des Spielers ruinieren kann…

Das Ziel ist hoch: man möchte Fußballer des Jahres werden. Zunächst sucht man sich einen passenden Verein aus den englischen Ligen, danach geht es der Kugel ans Leder – von wegen… Denn hier steht weniger der Sport als vielmehr das Geld im Vordergrund. Davon braucht man nämlich jede Menge, wenn man kicken will! Nur wer sich 'Torkarten' leisten kann, darf auch versuchen, Tore zu schießen. Vor jedem Spiel können ein bis drei dieser Karten, die jeweils mit einem bestimmten Manöver verbunden sind, eingesetzt werden. Die Manöver werden dann auf einer Tafel gezeigt und müssen vor dem gegnerischen Tor in der Vogelperspektive nachgespielt werden. Mit etwas Glück (oder Training) erzielt man dabei tatsächlich ein Tor (und etwas Geld), den Rest der Zeit verbringt man mit Bewunderung von Statistiken oder dem Warten auf zufällige Ereignisse wie Auf- oder Umstiegsmöglichkeiten. Eine gar seltsame Art von Fußball…

Man hat so den Eindruck, der einzige Zweck des Programms ist es, einen sensationell langsamen Algorithmus zum Zeichnen von Linien zu präsentieren.

Einen Vorteil hat das Spiel vielleicht doch: Man kann sich beim Gerangel um den Ball nicht verletzen. Dafür besteht jedoch ernsthafte Verletzungsgefahr, wenn man sich nach dem Kauf und erstem Probespiel 'sonstwohin' treten möchte…

(Michael Anton/hs)

AMIGA DOS Blitzlicht

Name: Footballer of the Year
Hersteller: Gremlin
Quelle: Fachhandel
Preis: zirka 64 DM

Positiv:
- die Anleitung eignet sich sehr gut zur Stabilisierung wackeliger Tische

Negativ:
- dafür ist der Preis jedoch zu hoch…



The Cycles

Wer mit den bislang für den Amiga erschienenen Motorradsimulationen noch nicht genug hat, kann sich jetzt mit The Cycles erneut auf den Balanceakt mit zwei Rädern begeben.

Das Rezept für die Entstehung von The Cycles war wohl folgendes: Man nehme ein Auto aus Test Drive II, halbiere es und gestalte die Fahrtstrecken in der Form eines Kreises. Tatsächlich erinnert vieles an dieses legendäre Straßenrennen, vor allem kleine Details wie der wählbare Schwierigkeitsgrad oder die Qualmwolke im (für ein Motorrad sehr unlogisch plazierten) Rückspiegel beim Überdrehen des Motors. Dies ist auch nicht verwunderlich, da für beide Spiele Accolade und Distinctive Software verantwortlich zeichnen. Allerdings geht es bei The Cycles nicht um geschicktes Ausmanövrieren von Cops, sondern um möglichst gute Rundenzeiten. Diese müssen gegen eine Vielzahl von Mitfahrern auf weltberühmten Rennstrecken in drei Hubraumklassen erkämpft werden.

Die Gestaltung des Rennens ist solider Durchschnitt, der zu Vergleichen mit anderen Programmen dieser Art reizt. Im Kontrast zu Super Hang On ist der Sound relativ mager, die grafische Gestaltung ist nicht ganz so liebevoll wie dort, aber wesentlich besser als in RVF Honda. Die von den Herstellern hervorgehobene einmalige 'Ego-Perspektive' ist jedoch nicht sonderlich sensationell, da bei The Cycles die Kamera lediglich etwas nach vorne gesetzt wurde. Der Sound beschränkt sich auf die Titelmusik und Motorengeräusche, die Steuerung ist akzeptabel. Der ultimative Fahrsimulator ist The Cycles wohl nicht.

(Michael Anton/hs)

AMIGA DOS Blitzlicht

Name: The Cycles
Hersteller: Accolade
Quelle: Fachhandel
Preis: 84,95 DM

Positiv:
- relativ gute Grafik
- wählbare Schwierigkeit

Negativ:
- nichts wirklich Neues



Gewinnen Sie mit Magic Bytes!

Nicht nur Computerspiele machen Spaß, dachten sich die Macher bei Magic Bytes und unsere Spieleredaktion. Warum nicht einmal ein richtig tolles Preisrätsel veranstalten, bei dem es auch etwas Tolles zu gewinnen gibt?

Gesagt, getan. Rund um den Bohrinselthriller 'North Sea Inferno', das neue, spannungsgeladene Magic-Bytes-Amiga-Spiel, haben wir uns ein kleines Bilderrätsel ausgedacht, das es zu knacken gilt. Wir haben auf dieser Seite drei verschiedene Bildschirmfotos zum Spiel 'North Sea Inferno' abgedruckt. Eines dieser drei Bilder stammt nicht aus diesem Spiel. Ihre Aufgabe ist es, herauszubekommen, welches der Bilder nicht paßt. Wenn Sie das richtige Foto gefunden haben, schreiben Sie dessen Nummer und Ihre Adresse auf eine Postkarte.

Unter den richtigen Einsendungen werden folgende Preise verlost:

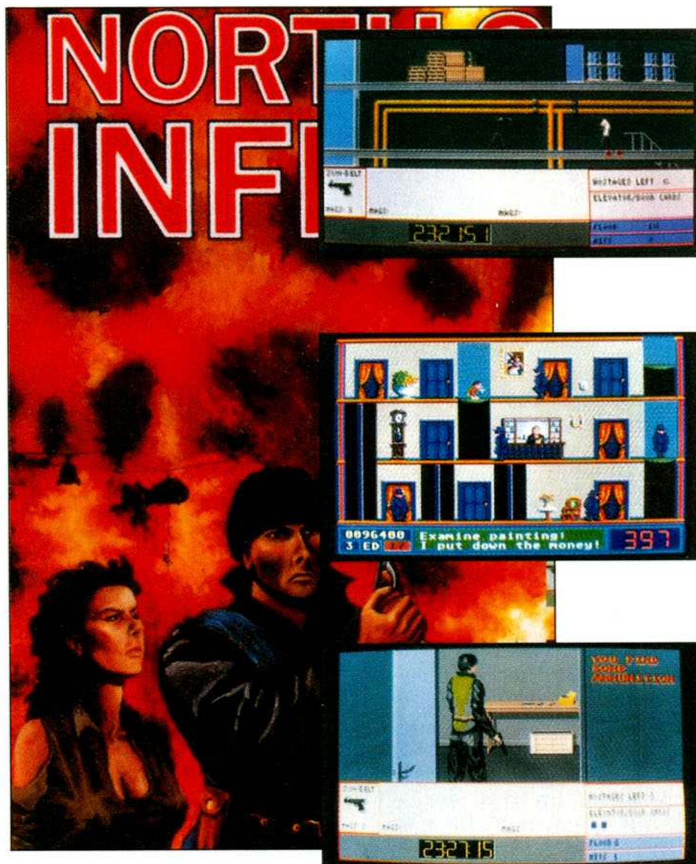
1. Preis: 1 Taschenfarbfernseher mit LCD-Flachbildschirm.

2. Preis: 3 Magic-Bytes-Spiele nach Wahl.

3. bis 5. Preis: Je ein Magic-Bytes-Spiel nach Wahl.

Ihre vollständig ausgefüllte Teilnahmekarte schicken Sie bitte an folgende Adresse:

DMV-Verlag
Red. AMIGA DOS
Kennwort Nordsee
Postfach 250
3440 Eschwege



Der Einsendeschluß ist der **08.05.1990**, es gilt das Datum des Poststempels. Mitarbeiter des DMV-Verlags und deren Angehörige sind von der Teilnahme ausgeschlossen. Der Rechtsweg ist ausgeschlossen. (hs)


Für Stryx sind die Roboter Freiwild – und umgekehrt!



In Stryx darf wieder zum Wohle und zur Rettung der Menschheit geballert werden. Diesmal jedoch, um ein selbst eingebrocktes Süppchen auszulöffeln, das mit einer Horde wildgewordener Androiden gewürzt ist...

Die Vorgeschichte zu Stryx ist zwar nicht gerade Pulitzerpreis-verdächtig, hebt sich jedoch etwas von den üblichen Legitimationen eines Gemetzels unter 'Un-Menschen' ab. In recht ferner Zukunft bedrohen ausnahmsweise keine Aliens die Menschheit, vielmehr wenden sich die Ergebnisse der modernen Zivilisation, hochgezüchtete Androiden, gegen ihre Erzeuger. Wie es dazu kam oder kommen wird? – Eine gute Frage...

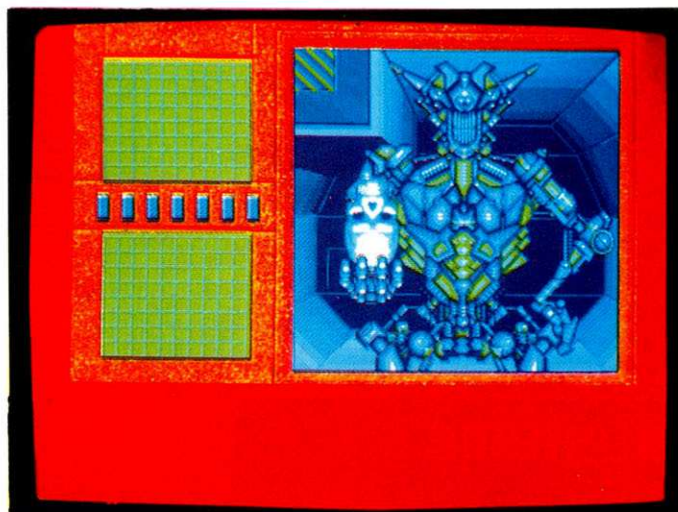
Wir schreiben das Jahr 3106, welches das Ende eines langen Krieges einläutet, der die Menschheit an den Rand des Ruins brachte. Das Leben ist nur noch in der Domstadt möglich, einem Zusammenschluß von fünf miteinander verbundenen 'Käseglocken'. Die Zivilisation stabilisiert sich wieder, im Laufe der Zeit erkennt man auch die Ursachen für den verheerenden Krieg: Eigentlich waren nur defekte Computer und Roboter am Chaos schuld. Also führt man ein Zeitlimit ein, nach dessen Überschreitung alle Computer und Roboter zerstört und durch neue ersetzt werden, bevor ihre künstliche Intelligenz zu gefährlich werden kann. Diese Aufgabe wird der 'Lifeforce', einem Kontrollmechanismus, anvertraut, der nur durch vier Schlüssel aktiviert werden konnte. Dieses System läuft einige Jahrhunderte lang recht gut, im Jahre 4516 kommt es jedoch erneut zu einer Rebellion der Roboter. Diese klauen die Schlüssel zur 'Lifeforce' und beginnen, alles Leben in der Domstadt zu zerstören...

Klar, daß hier ein erhabener Held helfen muß, ebenso klar, daß Sie als Spieler herhalten müssen. Der Retter heißt Stryx und ist halb Mensch und halb Roboter, 'the meanest fighting, smartest thinking machine ever invented'. Also den Ballermann geschultert und auf in den Kampf. Fünf Dome müssen nach den Schlüsseln für 'Lifeforce' und andere Türen durchsucht werden, in jedem Dom warten viele Roboter auf ihre Terminierung. Um von einem Dom in den anderen zu gelangen, müssen Kanäle durchquert werden, in denen es von Gegnern nur so wimmelt. Hilfsmittel gibt es genug – wenn man rechtzeitig an sie herankommt. In den Domen sind Waffen, Jetpacks und Energiereserven zu finden, in den 'Hives' unter den Domen kann man sich zusätzlich mit Power versorgen, allerdings geht es auch hier heiß her. Hat Stryx alle Schlüssel gefunden, braucht nur noch die 'Lifeforce' aktiviert zu werden, und der Spuk hat ein Ende...

Dies ist jedoch leichter gesagt als getan, denn der Schwierigkeitsgrad des Spiels ist irgendwo zwischen unmöglich und unfair. Dies beginnt mit der Vielzahl der Gegner, die den Helden mit Vorliebe von den Plattformen pusten (Strategie und Reaktion sind gefragt) und endet mit fehlenden Save- und Continue-Optionen und dem (sehr schnell verbrauchten) einzigen Bildschirmleben. Ob dies motiviert oder nicht ist reine Geschmackssache – die Vielfalt der Aufgaben und Schauplätze besitzt jedoch ihren Reiz. Vom technischen Standpunkt spricht jedenfalls wenig gegen das Spiel: Die Titelsequenz ist sehens- und hörenswert, der Sound im Spiel etwas dürftig, die Grafik insgesamt gelungen – lediglich die Steuerung in den 'Hives' vermag nicht zu überzeugen. Alles in allem verspricht Stryx viele Stunden Ballerspaß – vor allem, wenn man etwas masochistisch veranlagt ist. (Ich wünsche mir nichts sehnlicher, als einen soliden Cheat-Mode...)

(Michael Anton/hs)





AMIGA DOS Blitzlicht

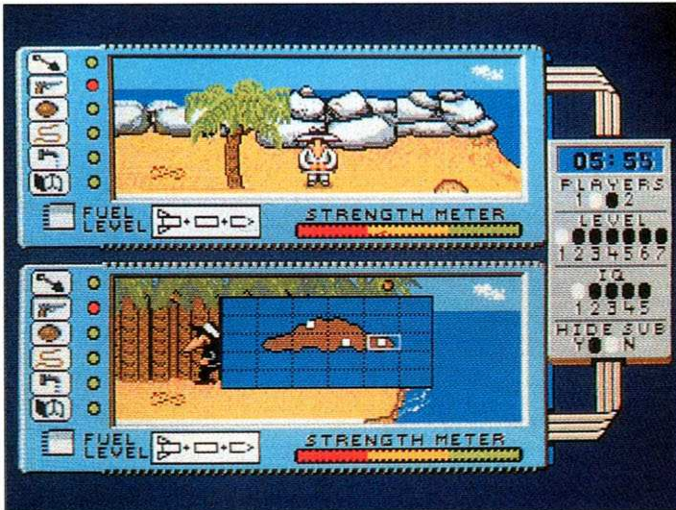
Name: Stryx
Hersteller: Psyclapse
Quelle: Fachhandel
Preis: zirka 70 DM

Positiv:
– viel Abwechslung

Negativ:
– sehr schwer!







Die MAD-Haus- und -Hofspione beim Einsatz auf einer einsamen Insel

Spy vs Spy – The Island Caper™

In dem satirischen Monatsmagazin MAD sind die beiden schon zur Institution geworden. Monat für Monat gehen der schwarze und der weiße Spion auf die Pirsch, um einander eins auszuwischen.

Irgendwann einmal wurde aus den beiden Spionen auch ein Computerspiel. Zur hohen Zeit des C-64 durften die beiden zwei Abenteuer bestehen. Nun liegt der leicht eingestaubte zweite Anlauf auch als Amiga-Spiel vor. Es geht darum auf einer einsamen Insel die versteckten Teile einer geheimen Rakete zu finden.

Beide Spione suchen zur gleichen Zeit die Insel ab und unterbrechen ihre Tätigkeit nur, um für den anderen eine teuflische Falle zu bauen oder ihm einfach mit einem Knüppel einen neuen Scheitel zu ziehen.

Spy vs. Spy kann alleine gegen den Computer oder mit einem zweiten menschlichen Teilnehmer gespielt werden. Für jeden der beiden Kontrahenten ist eine Hälfte des Bildschirms reserviert, in der sowohl ein Ausblickfenster als auch eine Liste der eingesammelten Gegenstände zu finden ist.

Seit ihrem ersten Auftritt auf dem C-64 haben die beiden Spione nur wenig dazugelernt. Spielprinzip und Musik lehnen sich eng am C-64-Original an, dementsprechend popelig und angestaubt präsentieren sie sich dann auch. Eigentlich hätte man aus den beiden Spionen ein feines Spiel machen können, doch was hier präsentiert wird, entstammt geradewegs dem Computermittelalter und ist irgendwo zwischen erträglich und ärgerlich anzusiedeln.

(hs)

AMIGA DOS Blitzlicht

Name: Spy vs. Spy
Hersteller: Wicked
Quelle: Fachhandel
Preis: 29,95 DM

Positiv:
- Zwei-Spieler-Option

Negativ:
- grauslicher Sound
- magere Grafik
- miese Animation





Der kleine Held, der prügelt munter, die Mutanten die Bunkertreppe runter

After the War

Nach dem Dritten Weltkrieg wird es duster um unsere ach so heiß geliebte Zivilisation.

Schenkt man der allgemein beliebtesten Endzeitvision Glauben, werden die wenigen, die den Krieg überleben, nichts Besseres zu tun haben, als einander das Fell über die Ohren zu ziehen.

So zieht ein weißblonder Jeansabenteurer durch die Ruinen einer Stadt, auf der Suche nach einem Bunkereingang. Denn während er durch die bedrohliche Landschaft zieht, ist sein Körper unentwegt der gefährlichen Strahlung ausgesetzt, die ihn irgendwann einmal hinraffen wird. Es sei denn, es gelingt ihm, zeitig genug den Bunker zu finden. Doch neben der Strahlung treiben sich noch tollwütige Hunde, brutale Kettenrocker, fiese Killerpunks und amoklaufende Excops in den Trümmern herum, und alle verstellen ihm den Weg in den Bunker. Glücklicherweise kann er unterwegs neben Waffen auch stärkende Getränke und Medikamente gegen die Verseuchung finden.

Vertikal scrollende Prügel-Epen sind seit Double Dragon nichts Neues. After the War ist zwar nicht in der Lage, das Thema originell zu variieren, aber dafür wird es in handfesten Genrebildern interpretiert. Es fehlen weder die Übermutanten am Ende jedes Levels, noch die in Trümmern liegende Freiheitsstatue in der Hintergrundgrafik.

Alles zusammengenommen ist das Spiel zwar solide gemacht, aber nicht gerade originell. Davon einmal abgesehen leidet das Spiel an demselben Übel, das auch alle anderen Vertreter des Genres plagt: Einmal durchgespielt ist schnell die Luft raus und die Spielmotivation sinkt rapide.

(hs)

AMIGA DOS Blitzlicht

Name: After the War
Hersteller: Dynamics
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- gute Grafik
- probate Steuerung

Negativ:
- wenig Neues
- wird schnell langweilig





Fordern Sie Sir Graham aus King's Quest zu einem Spiel Cribbage

Hoyle official Book of Games Volume 1

Im frühen 18. Jahrhundert faßte Edmond Hoyle die Grundregeln von Kartenspielen in einem Buch zusammen. Sechs dieser Spiele wurden in diesem Programm vereinigt, und als Gegner kann man zwischen richtigen Profis oder alten Bekannten aus den Sierra-Abenteuern wählen.

Karten zu spielen ist immer noch eine der beliebtesten Freizeitbeschäftigungen.

Man hat die Wahl zwischen: Crazy Eights, das etwa unserem Mau-Mau entspricht, Old Maid, unser Schwarzer Peter, Hearts, Gin Rummy und Cribbage, wo zusätzlich zu den Karten auch noch ein Spielbrett auf den Tisch gelegt wird, sowie Klondike Solitaire.

Nach der nun folgenden Gegnerwahl steht dem eigentlichen Spiel nichts mehr im Wege. Die Gesichter der Gegner sind allesamt animiert, und wenn Sie jemanden auswählen, erfahren Sie aus einer Sprechblase etwas über seine Geschichte und Spielstärke. Während des Spiels können Sie jederzeit die Regeln aus einem Menü heraus abfragen.

Hoyle Book of Games ist ohne Zweifel ein Spiel für die gesamte Familie, vom jüngsten bis zum ältesten Mitglied. Die guten Grafiken – der Schwarze Peter ist für die Kleinen sogar animiert – und die vielen unterschiedlichen Gegner verleihen diesem Spiel lange Zeit Spielspaß.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Hoyle official Book of Games
Hersteller: Sierra
Quelle: Fachhandel
Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- unterschiedliche Spiele mit verschiedenen Stärken
- gute Grafiken
- Regeln jederzeit abrufbar
- Save-Settings-Option

Negativ:
- Sound nervend, aber abstellbar
- komplett in Englisch





Mit dem Hubschrauber im Kampf gegen das Böse auf der Welt

FIRE!

Drogen, Terrorismus und Kriege sind allesamt Probleme, die den Frieden der Welt bedrohen. Mit dem Kampfhubschrauber Fire und dem Kriegsschiff USS New Deal als Basis können Sie diesen Problemen ein Ende bereiten. Also die Bordkanonen vorglühen und auf in den Kampf.

Es gilt, fünf gefährliche Aufträge zu lösen, von denen sich jeder in zwei Teile aufteilt. Da wären zum Beispiel der Drogenhandel in Kolumbien oder die Kämpfe im Libanon.

In Grünland hat eine Verbrecherorganisation ihr Hauptquartier aufgeschlagen und U-Boote zwischen Walen versteckt. Schließlich müssen Sie noch den Völkermord in Südostasien beenden und eine kriegerische Nation bändigen.

In den einzelnen Levels ist der Spielablauf immer gleich: Sie fliegen mit dem Hubschrauber durch eine horizontal in beide Richtungen scrollende Landschaft. Im oberen Bereich fliegen Hubschrauber, Flugzeuge und Raketen, während sich unten die eigentlichen Ziele befinden. Um ein Level zu beenden, muß man alles abschießen, was spätestens ab Level vier ziemlich schwer wird, da auch der Treibstoff ausgehen kann. Ab und zu verliert ein abgeschossenes Flugzeug einen Doppelschuß für den Hubschrauber oder eine Tonne mit zusätzlichem Fuel, die allerdings relativ schnell auf den Boden fallen. Das sind die einzigen Extrawaffen in diesem Spiel. Fire! ist ein Ballerspiel ohne besondere Extras oder spitzenmäßige Grafik. Daher ist es wohl eher in den mittleren Qualitätsbereichen anzusiedeln. Es ist aber leicht spielbar, und durch die Continue-Option motiviert es den Spieler ungemein, zumindest bis das Spiel geschafft ist.

(Robert Marz/hs)

AMIGA DOS Blitzlicht

Name: Fire!

Hersteller: New Deal

Preis: 89,– DM

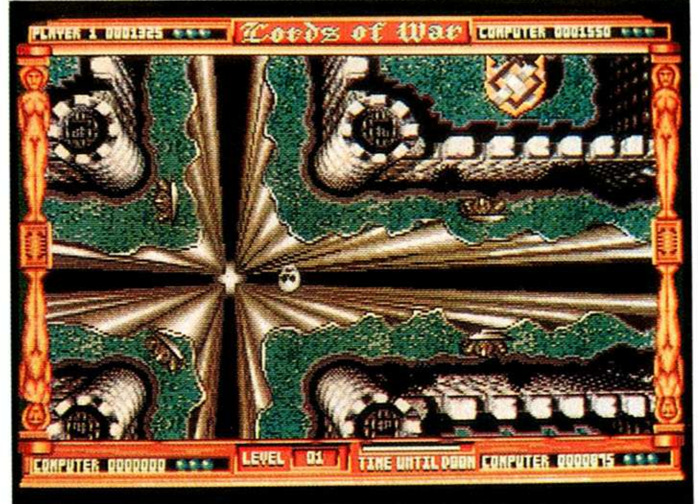
Positiv:
- verschiedene Aufträge
- Continue-Option

Negativ:
- wenig Extras
- kriegerische Handlung





Eins, zwei, drei, im Sauseschritt, rast Space Harrier über Amiga-Monitore



Breakout war einer der Ahnherren dieses Spiels, nur hier werden keine Steine zerschlagen, sondern Burgmauern

SPACE HARRIER II

Return to the Fantasyzone

Die Sonnenbrille vor die Augen geschoben, den tragbaren Laser unter dem Arm und die Jetboots an den Füßen, dürfen nun auch Amiga-Benutzer den zweiten Aufguß des Space-Harrier-Themas genießen.

Es gilt die verschiedenen Welten der Fantasyzone von eingedrungenen Aliens zu befreien. Während der Held die einzelnen Welten durchquert, versuchen die Biester ihn zu stoppen, indem sie sich mit vereinter Kraft, Welle für Welle auf ihn stürzen. Sein einziger Schutz ist die schwere, tragbare Laserkanone, die er bei sich hat. Die Zonen, durch die der Held muß, werden als dreidimensionale Szene dargestellt. Der Boden ist in ein Schachbrettmuster aufgeteilt und wird so animiert, daß der Spieler den Eindruck hat, über eine Ebene hinwegzuschweben. Unterdessen erscheinen die Ungeheuer aus der Tiefe des Raumes, oder aber im Rücken des Helden. Aus allen Rohren feuernd ziehen sie eine festgefügte Bahn durch das Szenario und verschwinden dann wieder, sofern sie bis dahin nicht schon abgeschossen wurden.

Space Harrier II ist bunt, laut und schnell. Leider ändern die Gegner ihre Angriffstaktiken im Laufe des Spiels kaum und der steigende Schwierigkeitsgrad kommt im Grunde nur durch verstärkte Alienattacken zustande.

(hs)

AMIGA DOS Blitzlicht

Name: Space Harrier II

Hersteller: Elite

Quelle: Fachhandel

Preis: 64,95 DM

Positiv:
- gute Musik

Negativ:
- wenig abwechslungsreich
- Gegner zu schnell

AMIGA DOS URTEIL: OK

GRAFIK	SOUND	MOTIVATION
7	8	6

Lords of War

Altbekannte Spielideen erleben immer wieder Neuauflagen. Und wenn das Spielkonzept so abgedroschen ist, daß es keinen Hund mehr hinter dem Ofen hervorlockt, dann wird es eben mit einer anderen Spielidee gekreuzt.

Ebenso wurde mit Lords of War verfahren. Breakout und Arkanoid sind die ursprünglichen Paten, denn es geht darum, einen Ball mit einem Schläger zu treffen.

Vier Burgen stehen an den vier Eckpunkten eines Rechtecks. In dem Raum zwischen den Mauern springt ein Ball hin und her. Vor jeder Burg ist ein Schild angebracht, der rund um den Wall bewegt werden kann. Wahlweise kann nun jede der vier Burgen durch einen menschlichen Spieler oder aber durch den Computer gegen den umherfliegenden Ball verteidigt werden. Dazu muß der Schild so vor der Mauer verschoben werden, daß der umherfliegende Ball durch den Schild umgeleitet wird. Trifft der Ball, der übrigens wie ein Totenschädel aussieht, die Mauer, schlägt er aus dieser ein kleines Stück heraus. Im Innenhof jeder Burg liegt ein Wappen, das durch die umgebenden Mauern geschützt wird. Zerschlägt der Ball die Mauer, gelangt in den Innenhof und trifft dort das Wappen, ist das Spiel für den Burgherren verloren.

Insbesondere daß vier Spieler gleichzeitig spielen können, verleiht diesem Spiel Reiz. Zaubersprüche und andere Extras machen aus Lord of War ein passables Geschicklichkeitsspiel, das auch nach längerem Spielen noch viel Spaß macht.

(hs)

AMIGA DOS Blitzlicht

Name: Lords of War

Hersteller: Digital Conceps

Quelle: Fachhandel

Preis: stand bei Redaktionsschluß noch nicht fest

Positiv:
- Vier-Spieler-Option
- Zaubersprüche

Negativ:
- Grafik wenig abwechslungsreich

AMIGA DOS URTEIL: GUT

GRAFIK	SOUND	MOTIVATION
7	6	8



Allerlei Horrorgetier erwartet Sie in Elvira, Mistress of the Dark


Hägar – Jetzt auch auf dem Amiga schrecklich aktiv!

Demnächst auf Ihrem Computer

Viele alte und neue Bekannte aus anderen Medien, wie zum Beispiel Comic und Film, erleben ihr Debut als Computerspielhelden. Es erwarten Sie Hägar, einer der schrecklichsten und beliebtesten Wikinger der Welt, und Elvira, die Lady der Finsternis und bekannte Komödien-Horror-Sexbombe amerikanischer Mitternachtsshows.

Elvira, Mistress of the Dark

Bisher durfte die attraktive Elvira sich nur im Firmensignet von Horrorsoft rekeln, jetzt hat sie ihr eigenes Horror-Rollenspiel, das geneigten Spielern mehr als einen Schauer über den Rücken jagen wird. Elvira hat das mittelalterliche Schloß einer ihrer Vorfahren geerbt, natürlich komplett mit allem möglichen unheiligen Monstergezücht. Was bisher an Grafik und Sound zu sehen und zu hören war, war sehr vielversprechend.

Hägar, der Schreckliche

Hägar, der beliebte Comic-Wikinger, erobert den Amiga. Auch in seinem Computerspiel darf Hägar seiner liebsten Beschäftigung nachgehen, ferne Länder zu plündern. Allerdings darf er nicht vergessen, seiner lieben Helga etwas mitzubringen. Und da es sich mit blanken Händen und leerem Magen schlecht kämpft, gilt es, sich zunächst mit Rüstzeug und Leckereien zu versehen.

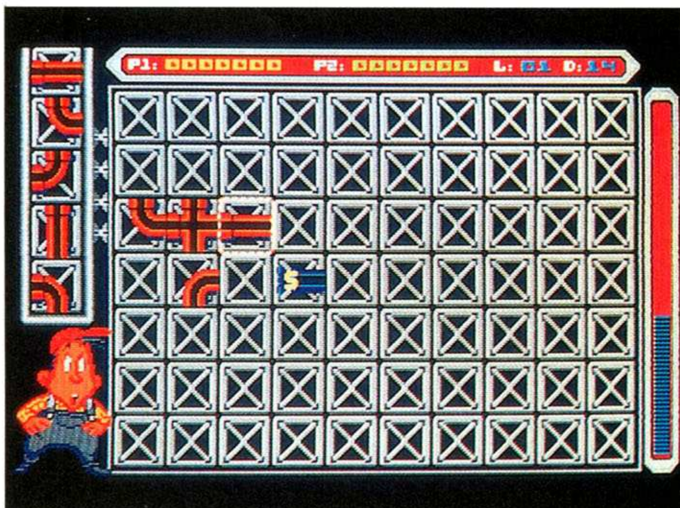
Pipe Mania

Mit diesem Titel steht uns ein feines Geschicklichkeitsspiel ins Haus, in dem es um Pipelines geht. Der Spieler muß aus verschieden geformten Teilstücken eine durchgehende Rohrleitung basteln. Nach einer gewissen Zeitspanne dreht jemand den Hahn der Leitung auf, und dann wird sich erweisen, wie gut das Rohr zusammengesetzt ist. Jede undichte Stelle rächt sich durch Punktabzug am endgültigen Score.

Venus

Ein neues, etwas anderes Ballerspiel wird Venus werden, von dem wir schon eine Demoversion spielen konnten. Ein insektenartiges Kunstwesen muß dabei versuchen, allerlei boshaften Monstern das Lebenslichtlein auszupusten. Neben Extrawaffen, guter Grafik und sauberem Scrolling erwarten den Spieler auch Zonen mit verkehrter Gravitation. Dann läuft Ihr Käferandroid auf dem Kopf stehend am oberen Bildschirmrand entlang.

(hs)


Pipelines zu bauen, ist nicht gerade leicht, besonders wenn mitten in der Arbeit die Schleusen geöffnet werden



Mal normal, mal auf dem Kopf, Venus präsentiert sich in opulenter Farbenpracht

Impressum

Herausgeber
Christian Widuch

Chefredaktion
Markus Matejka (mm)

Redaktion
Jürgen Borngießer (jb), Bernhard Rinke (br), Heinrich Stiller (hs), Joachim Freiburg (jf), Martin Schlöter (ms), Vera Brinkmann (vb)

Freie Autoren dieser Ausgabe
Edgar Meyzis, Bernd Rudolf, Robert Marz, Jürgen Seibel, Garry Glendown, Ulf Petersen, Michael Anton, Andreas Polk, Sebastian Ritter, Stefan Hochmuth, Ralf Zuber, Willi Bäcker, Monika Schmid

Redaktionsassistenz
Anke Kerstan (ke), Susanne Eska (es)

Koordination
Stefan Ritter

Produktionsleitung
Gerd Köberich

Bereichsleitung
Claudia Ebbrecht (Fotosatz/Lektorat), Margarete Schenk, Helmut Skoupy (Montage/Reprografie)

Layout
Michael Grebenstein

Fotografie
Christian Heckmann

Fotosatz
Gabriela Joseph

Lektorat
Susanne Lessinger

Montage/Reprografie
Manuela Eska, Peter Gajewski

Werbegestaltung
Mohamed Hawa, Petra Küch

Anzeigenverkaufsleitung
Wolfgang Brill

Anzeigenverkauf
DMV-Verlagsbüro München
Zaunkönigweg 2c, 8000 München 82
Telefon (0 89) 4 39 10 87, Telefax 0 89/4 39 10 80
Leitung: Britta Fiebig
Anzeigenverkauf: Monika Schöbel, Jens Dhein, Peter Schätzle

Anzeigenpreise
Es gilt die Anzeigenpreisliste Nr. 1 vom 01.01.1990

Anschrift Verlag/Redaktion:
DMV Daten & Medien-Verlag
Widuch GmbH & Co. KG
Fuldaer Straße 6
3440 Eschwege, Telefon (0 56 51) 8 09-0,
Telefax (0 56 51) 8 09-3 33

Vertrieb
Verlagsunion Erich Pabel-Arthur Moewig KG (VPM), Friedrich-Bergius-Straße 20, 6200 Wiesbaden

Druck
Druckerei Jungfer, 3420 Herzberg

Bezugspreise
»AMIGA DOS« erscheint monatlich.
Einzelpreis DM 6,50/sfr. 6,50/öS 52, –

Abonnementpreise
Die Preise verstehen sich grundsätzlich einschließlich Porto und Verpackung.

Inland:
12 Ausgaben: DM 70, –
6 Ausgaben: DM 35, –
Europäisches Ausland:
12 Ausgaben: DM 100, –
6 Ausgaben: DM 50, –
Außereuropäisches Ausland:
12 Ausgaben: DM 120, –
6 Ausgaben: DM 60, –

Bankverbindungen:
Postscheck Frankfurt/M: Kto.-Nr.: 23043-608
Raiffeisenbank Eschwege:
BLZ: 522 603 85, Kto.-Nr.: 245 7008

Die Abonnementbestellung kann innerhalb einer Woche nach Auftrag beim DMV-Verlag, Postfach 250, 3440 Eschwege, schriftlich widerrufen werden. Zur Wahrung der Frist reicht der Poststempel. Das Abonnement verlängert sich automatisch um 6 bzw. 12 Monate, wenn es nicht mindestens 6 Wochen vor Ablauf beim Verlag schriftlich gekündigt wird.
Für unverlangt eingesandte Manuskripte und Datenträger sowie Fotos übernimmt der Verlag keine Haftung. Die Zustimmung zum Abdruck wird vorausgesetzt.
Eine Haftung für die Richtigkeit der Veröffentlichungen kann trotz sorgfältiger Prüfung durch die Redaktion vom Herausgeber nicht übernommen werden. Die geltenden gesetzlichen und postalischen Bestimmungen sind zu beachten.

Namentlich gekennzeichnete Fremdbeiträge geben nicht in jedem Fall die Meinung der Redaktion wieder.

Die Inserenten

Im nächsten Heft

Mit BTX geht's fix! Wer ein Postmodem für BTX hat, der braucht nur noch die passende Software und ein entsprechendes Kabel. Wir haben uns in BTX umgesehen.



Bild 1. Digi-Tiger gegen den Rest der Welt: Wir haben uns den handlichen Digitizer einmal für Sie angesehen. Was er alles zu leisten vermag, lesen Sie in der nächsten Ausgabe

386er Power: Von der Firma Roßmöller gibt es eine Neuheit für PC-Karten-Besitzer: eine Erweiterungskarte mit einem 386SX-Prozessor. Ob der Amiga damit zum Super-PC wird, erfahren Sie im nächsten Heft.

Amiga und Drucker – ein nicht immer unproblematisches Thema. Wir haben den Star LC24-15 einem Test unterzogen. Das Ergebnis finden Sie in AMIGA DOS 6/90.

Das haben Sie schon immer gesucht: Jede Menge neuer Tips & Tricks zu Amiga-BASIC, CLI, Assembler, Modula2, C, GFA-BASIC warten auf Sie.

Topaktuell... Die neuesten Spiele auf dem Softwaremarkt haben wir für Sie gesichtet. Außerdem wieder jede Menge Tips zu den beliebtesten Spielen auf dem Amiga in unserer Helpline.

Und natürlich... eine geballte Ladung Listings, Tests und Infos.

DFÜ – Begleiten Sie uns auf einem ausgedehnten Streifzug durch die Welt der Datenfernübertragung. Ob Modems, Koppler oder Übertragungsprotokolle, AMIGA DOS hat alles Wissenswerte über Hard- und Software zu diesem Thema für Sie zusammengestellt. Außerdem: eine Liste von interessanten Mailboxen.

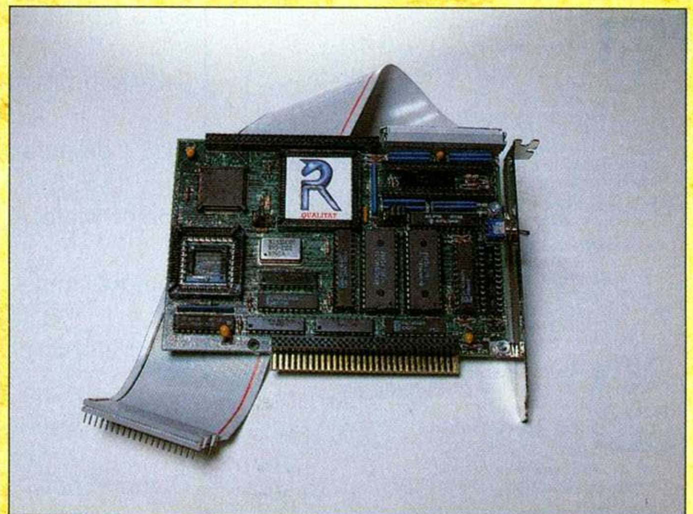


Bild 2. Die 386SX-Karte – Lohnt sich der Einbau?

Die nächste

AMIGA DOS

finden Sie ab

09. Mai '90

bei Ihrem Zeitschriftenhändler



Bild 3. Venus: Kampf den Käfern ist der Inhalt dieses sehenswerten Shoot'em Ups. Gute Grafik und reichlich Action – mehr in der nächsten AMIGA DOS

STEFAN OSSOWSKI'S SCHATZTRUHE

(1) **Haushaltsbuch** bis zu 25 Konten, flexibel, leicht bedienbar, mit umfangreicher **deutschsprachiger** Dokumentation DM 8,-

(3) **MountainCAD** professionelles CAD-Programm, **deutsche Anleitung** DM 8,-

(4) **Spiele I, II, III** 10 erstklassige PD-Spiele aus allen Bereichen wie Action, Geschicklichkeit, Strategie (3 Disketten) DM 24,-

(5) **Anti-Virus** 8 Programme gegen alle Viren DM 8,-

(6) **Text** hochwertige **deutsche** Textverarbeitung DM 8,-

(7) **Utility-Disk** 25 nützliche Utilities aus allen Bereichen DM 8,-

(9) **Sonix-Paket** Original-Sonix-Player + 4 weitere Disketten mit phantastischer Sonix-Musik. **Top-Hit!** DM 40,-

(10) **Business** 3 Disketten: Tabellenkalkulation, relat. Datenbank, sehr gute Textverarb. Vers. engl. DM 24,- Vers. deutsch DM 70,-

(13) **Paranoid** sensationelles Breakout-Spiel DM 8,-

(14) **Buchhaltung** erstes **deutsches** PD-Buchhaltungsprogr. DM 8,-

(16) **AMIGA-Paint** sehr gutes **deutsches** Malprogramm DM 8,-

(17) **Videodatei** bringt Ordnung in Ihre Videodatei, **deutsch** DM 8,-

(18) **Fußballmanager** bei diesem Spiel können Sie Ihre Fähigkeiten als Manager eines Fußballclubs testen, **deutsch** DM 8,-

(20) **Giroman** komfortables **deutsches** Programm, mit dem Sie Ihr Girokonto einfach verwalten können DM 8,-

(22) **Kampf um Eriador, V 2.0** taktisches Strategiespiel für 2 Personen mit sehr guter Grafik und Sound, **deutsch** DM 8,-

(26) **Risiko** die Amiga-Umsetz. d. bek. Brettspiels, **deutsch** DM 8,-

(27) **Ray-Tracing-Construction-Set, V 2.0** phantastisches Programm zur Berechnung von Licht und Schatten - siehe Test Amiga 1/88 - komplett auf **3** Disketten mit **deutscher Anleitung** DM 24,-

(28) **Wizard of Sound** ein phantastisches Musikprogramm zur Erstellung eigener Lieder, mit **deutscher** Anleitung (2 Disks) DM 10,-

(29) **Broker** ein sehr gutes **deutsches** Börsenspiel DM 8,-

(30) **Quickmenü** erst. Sie sich Ihre eig. Workbench i. **deutsch** DM 8,-

(31) **Blizzard** phantastisches Ballerspiel m. sehr guter Animat. DM 8,-

(32) **DSort deutsches** Diskettenkatalogisierungsprogramm DM 8,-

(33) **Pascal** ein komplettes Pascal-Paket (3 Disketten) mit Compiler, **deutscher Anleitung** u. einem s. gut. deutschen Editor DM 24,-

(34) **DiskKey** Diskettenmonitor mit **deutscher Anleitung** DM 8,-

(35) **Peters Quest** Geschicklichkeitsspiel mit lustiger Handlung und **deutscher Anleitung** DM 8,-

(36) **Spiele** auf dieser Diskette sind 3 Tetris-ähnl. Spiele enth. DM 8,-

(37) **MRBackup** Festplattensicherungsprogramm mit **deutscher Anleitung** DM 8,-

(38) **Universal-Datei deutsches** Datei-Verwaltungsprogr. DM 8,-

(39) **Assembler** ein komplettes Entwicklungssystem für Maschinensprache in **deutsch!** DM 8,-

(40) **Bibel-Quiz** lehrreich und unterhaltsam DM 8,-

(42) **BootMaster** Mit diesem Programm können Sie individuelle Bootblöcke mit Lauftext und Sternenhintergrund erstellen. DM 10,-

(43) **Banner II** Dieses Programm ermöglicht es Ihnen, komplette Banner mit Ihrem Drucker zu erstellen. Leicht bedienbar! DM 8,-

(44) **Boulder V1.3** Boulder ist ein **sehr schnelles Geschicklichkeitsspiel**, das an den C 64-**Klassiker Boulder-Dash** angelehnt ist. Ohne Sound! DM 8,-

(45) **Label-Paint** deutsches Etikettendruckprogr. mit Grafik! DM 8,-

(46) **Roll On** friedliches Geschicklichkeitsspiel mit Leveleditor. Ein Spiel mit langanhaltender Motivation. Super! DM 8,-

(47) **Paccy** der alte Spielhallenklassiker lebt wieder auf DM 10,-

(48) **Pente** Ein intelligentes "5-Gewinnt"-Spiel. **Deutsch!** DM 10,-

(49) **Tumbler Street** Glücksspiel, bekannt durch **Salvatore** von **RTL-Plus!** Benötigt 1 MB Speicher! DM 8,-

(101) **RIM-5 = Relationale Datenbank**
Äußerst leistungsfähig, sowohl für den privaten als auch für den geschäftlichen Bereich geeignet. **Mit deutscher Anleitung und ausführlichem Einführungskurs.** DM 30,-

(102) **AnalytiCalc = Tabellenkalkulation**
Leistungsstark mit deutscher Anleitung. Ein unverzichtbares Hilfsmittel für Kalkulationsaufgaben jeder Art (**Bericht Amiga 12/89**). DM 30,-

(103) **DEA Arithmetica = Die Göttin Arithmetica**
Besonders für Schüler, Studenten und Lehrer geeignet. Ableitungen, **Kurvendiskussion** und Skizzieren von Funktionsgraphen problemlos möglich. **Deutsch!** DM 30,-

(104) **Haushaltsbuch Version 2.1**
Komplettlösung zur Verwaltung Ihrer privaten Finanzen! Leicht bedienbar, auch für Computer-Neulinge geeignet. Unterstützung durch Demo-Dateien, frei von buchhalte-rischen Fachausdrücken. Funktionsüberblick: verschiedene Dateien, mehrere Kontenlisten, bis zu 10 Bilanzen, 40 freidefinierbare Konten, Suchroutinen, doppelte Buchführung, Filterfunktionen, Jahresübertrag, Mausunterstützung ... **Deutsche Dokumentation im Ringbuch!** 1 MB Speicher erforderlich! DM 98,-

(105) **Xytronic II**
Intergalaktische Handelssimulation mit guter Grafik und gutem Sound sowie sehr hoher Motivation. **Natürlich in Deutsch!** DM 29,-

(106) **Data-Manager**
Eine sehr leicht zu bedienende Dateiverwaltung, mit der Sie Adressen, Mitgliederlisten etc. perfekt verwalten können. **Deutsches Programm mit deutscher Anleitung!** DM 15,-

(108) **Bundesliga 2000**
Programm zur Bundesligaergebnisverwaltung und Tabellenberechnung. Verwaltung beliebig vieler Ligen möglich. Historie seit 1985. **Deutsch!** DM 15,-

(109) **Money Player Deluxe – Geldspielgerät**
Sie fühlen sich wie in der Spielhalle! Start- und Risikoautomatik, Sonderspiele viele Extras und toller Spielspaß. Palauflösung, Maussteuerung! DM 39,-

(110) **Wizard of Sound 2.0 – Musikprogramm**
WoS 2.0 ist ein sehr gutes Musikprogramm mit 61 Instrumenten, kompletter Notation, Player-Programm, Instant-Replay-Modus, einfachste Noteneingabe, ... Auch für Musik-Laien geeignet! **Auf 2 Disketten mit deutscher Dokumentation!** DM 35,-

(111) **Broker 1.2 – Das Börsenprogramm**
Sehr **realistische Börsensimulation,** mit bis zu 4 Spielern spielbar. Broker 1.2 ist unterhaltsam, spannend und lehrreich. **Deutsch!** 1 MB Speicher! DM 29,-

(112) **Vokabel-Trainer-Englisch 1.0**
Leicht zu bedienendes, leistungsfähiges Lernprogramm. Der Wortschatz ist individuell erweiterbar. Damit macht das Lernen Spaß! **Deutsch!** DM 15,-

(114) **BootMenü**
Sie können im Disketten-Bootblock ein Menü einrichten, über das Sie beim Bootvorgang per Maus 10 verschiedene Programme aufrufen können! **Mit deutscher Anleitung!** DM 19,-

(115) **CBB – CopyBootBlock Installierungsprogramm**
Mit CBB können Sie in Ihrem Bootblock ein Kopierprogramm installieren, das sie bei **jedem Bootvorgang per Mausklick starten können.** 2 Laufwerke oder 1 MB. DM 19,-

(116) **Danger Castle**
Ein Hüpf- und Rennspiel für 1 oder 2 Personen mit Wahnsinnsgrafik, Stereosound und 20 verschiedenen Levels! Deutsche Anleitung! DM 39,-

(117) **SpeedRunner**
Eine Loadrunner-Variante mit Level-Editor, 100 Levels, Super-Grafik und Klasse-Sound. In Assembler geschrieben und Palauflösung! Deutsche Anleitung! DM 39,-

(119) **Einkommensteuer '89**
Mit diesem Programm können Sie sehr einfach Ihre Lohn- und Einkommensteuer berechnen. Deckt 99 % aller Fälle ab. Es werden viele Sonderfälle behandelt. **Deutsch!** DM 79,-

(120) **Chemie auf dem Amiga**
Ein didaktisch hervorragend aufgebautes Lernprogramm für die Klassen 7-11. Einfachste Bedienung, viele erläuternde Grafiken und ein breites Themenspektrum zeichnen dieses Programm aus. DM 49,-

(121) **Tape It!**
Ein professionelles Programm zur Verwaltung Ihrer Musiksammlung. Vielfältige Listen können erstellt und Kassettenhüllen bedruckt werden. Deutsch! DM 19,-

(122) **SchreibM 2.0 – Schreibmaschinentrainer**
Spielend lernen Sie Ihre Tastatur in den Griff zu bekommen. DM 19,-

(123) **K.A.L.-Verwaltung**
Einfache Kunden-, Artikel- und Lieferantenverwaltung DM 19,-

(124) **SGM – Statistik-Grafik-Manager**
Auf einfache Art und Weise können Sie mit **SGM** Statistik- und Präsentationsgrafiken erstellen: Balken-, Torten-, Tendenz-, Flächengrafiken. Diese Grafiken können ausgedruckt oder im IFF-Format weiterbearbeitet werden. **Deutsch!** DM 49,-

(125) **Lotto V.1.01**
Auf dieses **deutsche** Programm haben Lotto-Fans schon lange gewartet. Mit **Lotto V.1.01** können Sie Ergebnisse verwalten, Tips erstellen, Statistik betreiben, u.v.m. ... Mit VEW-System! DM 19,-

(126) **AmigaDraw V.1.2**
Ein professionelles deutsches Mal- und Zeichenprogramm DM 19,-

(127) **Hawk Inc.**
Ihre Aufgabe ist es, einen schweren Fall von Wirtschaftskriminalität aufzudecken! Dringen Sie in den Zentralcomputer ein und lösen Sie den Fall! **Ein realistisches und spannendes Spiel!** DM 19,-

(128) **Aktien-Depot-Manager**
Eine Aktien-Verwaltung par Excellence! Mit grafischer Chart-Darstellung, Depot-Management, vielen Optionen, chartanalytische Kaufempfehlungen, etc. ... **Mit umfangreicher deutscher Anleitung!** DM 49,-

(201) **CLI-Help-Deluxe Software für Einsteiger!**
Mit dieser Tutorial-Diskette lernen Sie das CLI kennen und beherrschen! Speziell auf die Bedürfnisse von **Anfängern und Einsteigern** ist diese hochinformative Diskette abgestimmt. Mit dieser Diskette gehören Probleme mit Amiga-DOS der Vergangenheit an! DM 19,-

*** TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN * TAIFUN**
Die deutsche Public-Domain-Serie aus dem Hause Ossowski! Wußten Sie schon, daß beim PD-Versand Stefan Ossowski alle drei Monate 10 neue TAIFUN-Disketten mit den interessantesten Neuerscheinungen des PD-Marktes vorgestellt werden? Bevorzugt präsentieren wir Ihnen die neuesten deutschen Programme. Am 15. Februar erschienen die neuen TAIFUN-Disketten Nr. 121 bis 130.
Schnupperpreis: DM 53,- V-Scheck, DM 57,- Nachnahme

Versandkosten	**Inland:**	**DM 3,- V-Scheck**	**DM 7,- Nachn.**
(Porto/Verpackung):	**Ausland:**	**DM 6,- V-Scheck**	**DM 15,- Nachn.**

ABO-SERVICE
Bei uns erhalten Sie fast jede PD-Serie auch im günstigen Abonnement! Auf unsere Staffelpreise gewähren wir außerdem einen **10%igen ABO-Rabatt**! Die Fish-Serie ist z. B. schon bis Nr. 310 lieferbar! Rufen Sie uns doch einfach an oder schreiben Sie uns, wenn Sie an weiteren Informationen zu unserem ABO-Service interessiert sind!

Zuverlässigkeit
+ Schnelligkeit
+ Service
= PD-Versand Stefan Ossowski

Testen Sie uns!

Wir führen alle bekannten PD-Serien wie Fish, Panorama, Faug Amicus, Auge, Taifun, Chiron, RPD, Kickstart, Slideshows, TBAG, Franz ... Viele Programme deutschsprachig.

5,-DM

kosten unsere aktuellen **2** Katalogdisketten. Mit deutschem Inhaltsverzeichnis unseres gesamten PD-Angebots von weit über 1500 Disketten. Ab DM **4,50**. Gegen Verrechnungsscheck oder in Briefmarken anfordern.

Stefan Ossowski - Entwicklung und Vertrieb von Software und Public Domain
Veronikastraße 33, D-4300 Essen 1, Tel./Btx: 0201/788778


DMV
Daten- und Medienverlag